BALANCING AND GRASPING FROM VISUAL FEEDBACK FOR AN UNSTABLE WHEELED HUMANOID

A Dissertation
Presented to
The Academic Faculty

By

Areeb A. Mehmood

In Partial Fulfillment
of the Requirements for the Degree
Masters of Science in the
Woodruff School of Mechanical Engineering

Georgia Institute of Technology

December 2019

BALANCING AND GRASPING FROM VISUAL FEEDBACK FOR AN UNSTABLE WHEELED HUMANOID

Approved by:

Dr. Hutchinson, Thesis Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Rogers, Member
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Ueda, Member
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: August 19, 2019

Areeb A. Mehmood

Dedicated to my mother and father.

ACKNOWLEDGEMENTS

I would like to first acknowledge Munzir Zafar for adopting me in the lab to work on this amazing machine, Golem Krang. Thank you for teaching me everything about the robot, all of the myriad fields related to working in robotics, and even the more mundane aspects of maintaining hardware. All of this opened up the opportunity for me to do my own work in the lab. More importantly, thank you for providing the life mentorship I never knew I was missing. I have truly appreciated your perspectives throughout the past two years.

Secondly, I must thank Professor Seth Hutchinson for advising my thesis, and guiding me to do work I found exciting. Your insight through the process was very valuable, and your appreciation of my efforts encouraged me to continue working through the harder problems in this project. Thank you to Professors Jun Ueda and Jonathan Rogers for serving on my Reading Committee, as well as the greater Georgia Tech faculty that assisted me throughout the process.

Thank you to my close friend Hadi Habib, for the reassurances from your own graduate school experiences, and the conversations and companionship you provided even though you were far away. Thank you to my Mamoo and Mani, who have always encouraged my efforts and guided me to the better path. Thank you to my siblings, Roasa, Shikeba, and Hevatib. For a lifetime, I have turned to each of you for the unique qualities each of you brings. You posses unique places in my heart. And finally, to the two individuals I depend on for immeasurable and unwavering love which carries me farther than I could go myself, my dear parents, to whom I dedicate my life's work.

TABLE OF CONTENTS

Acknowledgments v

List of Tables ix

List of Figures x

Chapter 1: Introduction and Background 1

1.1 The Robotic Platform 2

1.1.1 Hierarchical Control Scheme and MPC 3

1.1.2 Whole Body Control and Tasks 3

1.1.3 Krang Capabilities 4

1.2 Research Objectives 4

1.2.1 End Effector Control 5

1.2.2 Visual-Servo Control 5

1.2.3 Base Angle Approximation 6

1.2.4 Localization 6

Chapter 2: Review of Previous Work 8

2.1 Visual Servo 8

2.2 Localization and Base Angle Approximation 9

Chapter 3: Technical Approach 11

3.1 Selection of Sensor 11

3.1.1 RGB-D vs Stereo 12

3.1.2 ZED Camera Features 13

3.2 Positional Tracking 14

3.3 End-Effector Control 15

3.3.1 Inverse Kinematics Formulation 17

Chapter 4: Object Detection and Localization 19

4.1 Obtaining and Filtering the Target Object Position 19

4.1.1 Detecting the Object 19

4.1.2 Obtaining the Object Position 22

4.1.3 Workspace for Visual Servoing 25

Chapter 5: End-Effector Control 27

5.1 Optimization-Based Inverse Kinematics Formulation 27

5.2 Hardware Control with Linear Interpolation 34

Chapter 6: Base Angle Approximation and Localization 39

6.1 Generating Smoother IMU Data 39

6.2 Mathematical Description of Transformations 40

6.3 Generating an Area File for the Experiment Environment 42

6.4 Localizing 43

Chapter 7: Results 44

7.1 Visual Servoing 44

7.2 Base Angle Approximation 47

7.3 Localization 48

Chapter 8: Future Work and Conclusion 49

8.1 Future Work 49

8.1.1 Position-Based Visual Servoing 49

8.1.2 Obstacle Avoidance/Safety Guarantees 49

8.1.3 Advanced Vision 50

8.2 Limitations 50

8.3 Conclusion 51

Appendix A: IMU Plots 54

References 58

LIST OF TABLES

3.1 Parameters Set on ZED Camera 14

4.1 Range of HSV Values 21

4.2 Mean, Variation and True Value of Detected Positions (cm) 24

5.1 Parameters for IK Simulation 31

5.2 Differences in Maximum Joint Velocities (rad/s) 38

LIST OF FIGURES

1.1 Golem Krang Hardware 2

1.2 Krang Whole Body Control Scheme 3

3.1 ZED Stereo Camera (Source: StereoLabs) 13

3.2 ZED Camera on 45° Mount 14

3.3 Transformations for a Single Arm of Krang 16

3.4 System Architecture 18

4.1 Raw Image with Detected Region Wrapped in a Contour 20

4.2 HSV Map of the Raw Image (top) Mask of the Detected Region (bottom) . 21

4.3 Raw Image (top) and Corresponding Depth Map (bottom) 23

4.4 Position of Static Tennis Ball (100 steps = 7.5 seconds) 24

4.5 Positional Data Over Time with Running Average Smoothing (100 steps ≈ 7.5 seconds) 25

4.6 Viewing Space of the Camera 26

5.1 Joint Movement and Changing End Effector Error 32

5.2 Joint Movement and Static Starting End Effector Error 32

5.3 Visual of Movement for the Plot of Figure 5.2 33

5.4 Plot of Hardware with Changing Error 35

5.5 Initial Error with no Interpolation (Simulation) 37

5.6 Initial Error with Interpolation (Simulation) 37

6.1 Plot of Raw IMU Data (blue) with Filtering (orange) 40

6.2 Krang Side View In Sitting and Standing Configurations (No Arms) 41

7.1 Plot of Position of End Effector and Target Object Over Time 44

7.2 Arm Following a Moving Target Object 45

7.3 Arm Moving to Stationary Target from Resting Position 46

7.4 Base Link Position from IMU and Camera Data 47

7.5 ZED Camera Tracking its Location around the Lab 48

A.1 High Range IMU Data 54

A.2 More Detailed IMU Data (rad vs s) 55

SUMMARY

Krang is a Wheeled Inverted Pendulum Humanoid, designed to accomplish strenuous tasks quicker, and with more strength, than the average human being. Weighing over 300 lbs, Krang sits on a differential drive platform balancing on two wheels in an inverted pendulum configuration. The platform forms the first joint in a 17 degree-of-freedom upper body that possesses a waist, torso and two 7 degree-of-freedom arms. Through a whole body control scheme, this unique design allows Krang to manipulate its center of mass to locomote quickly on a plane, while the redundancy of joints enables second order tasks to be completed, such as carrying a tray of water or utilizing its weight torque to lift and move heavy objects. However, while Krang is very capable, it remains unaware of the environment in which it works.

This research project aims to introduce localization and state estimation capabilities to Krang by giving it the ability to measure and analyze its surroundings. Currently, Krang must be positioned by humans before running experiments involving locomotion and end effector manipulation, making the robot blind to variations in its environment, and vulnerable to potentially poor state estimation of the first link. By attaching a vision system to the robots spine, this thesis project aims to introduce positional tracking and spatial mapping capabilities, which can act as a redundancy for stabilization of the robot, and give Krang a level of autonomy that requires less human oversight. In addition, a visual servoing formulation allows the robot to identify and pick up objects on its own.

CHAPTER 1
INTRODUCTION AND BACKGROUND

In order for an autonomous robot to fulfill its design objectives, it must be able to understand its environment and correct itself in the event of drifts or dynamic changes in the environment, to perform its tasks for long periods of time with little or no human oversight and intervention. This is particularly true for robots that are large and unstable, or perform safety critical tasks such as rescue operations. In [1], an overview of current robotic platforms reveals that there is ongoing work for the development of autonomous rescue robots that vary in size and objectives.

Some specific capabilities these robots have is the ability to localize and execute loop closure in a known environment, to avoid obstacles and map unknown environments, to be able to utilize faculties such as grasping, pushing and climbing by analyzing the environment, and to utilize intelligent capabilities for tertiary goals such as safety. As such, this work explores the usage of an environment-sensing technology to accomplish many of these tasks for an inherently unstable, inverted pendulum humanoid robot. This is accomplished by localizing the robot in a known environment, locomoting to pre-allocated positions or finding new ones, and by allowing the robot to grasp recognized objects while maintaining a stable pose.

The list of contributions in this project are:

- Development of an inverse kinematics controller for the robot's arms and deployment of the controller on hardware
- Estimation of base link state from visual data
- Implementation of OpenCV to detect objects of interest
- Localization of the robot in a known environment

Figure 1.1: Golem Krang Hardware

1.1 The Robotic Platform

Golem Krang is a two-wheeled, inverted pendulum humanoid robot. It possesses a differential drive system, upon which is mounted a waist and torso joint. Upon the torso are connected two shoulders that form the bases of two seven degree-of-freedom (DOF) arms (Figure 1.1). One current project of the lab is the development of a control algorithm which will allow the robot to plan locomotion on a plane while accomplishing tasks with the arms, such as carrying a tray of water, or using the huge leverage of the robots weight to carry and move heavy objects.

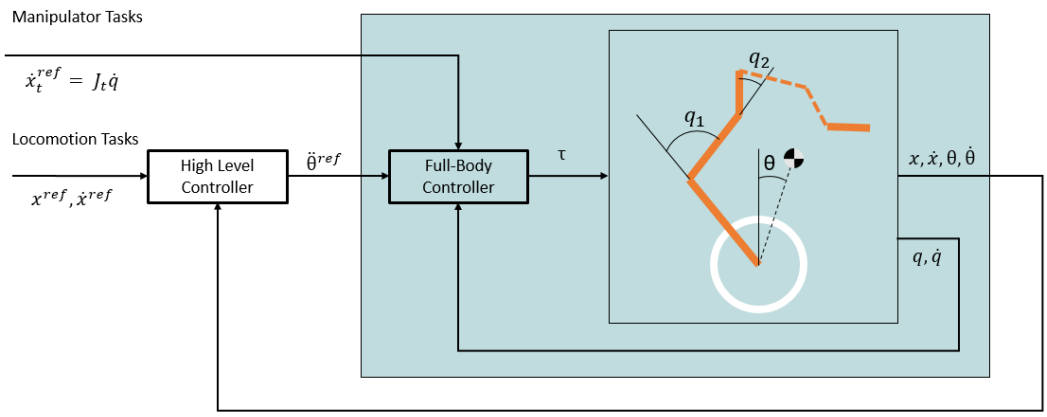

Figure 1.2: Krang Whole Body Control Scheme

1.1.1 Hierarchical Control Scheme and MPC

In order to locomote while performing tasks with the upper body, a hierarchical control framework is deployed on Krang, which allows for the upper body to participate in meeting center of mass (COM) trajectory targets for locomotion, while taking advantage of the highly redundant structure of Krang to simultaneously accomplish other tasks at its end effectors. The high level controller in this framework uses model predictive control to generate these COM trajectories. The low level controller in this hierarchy manipulates the upper body of Krang to meet the COM requirements of the high level controller while accomplishing end effector tasks [2].

1.1.2 Whole Body Control and Tasks

Whole Body Control refers to a control scheme in which the robotic system accomplishes a task by simultaneously controlling the joints in the entire body. In previous work using a naive control scheme, any movement in the upper body of Krang resulted in the robot exhibiting a ”knee-jerk” reaction to stabilize the entire platform, resulting in loss of end-effector placement. This was due to the fact that the high-level controller was completely blind to what the upper body was doing. However, after successful implementation of the Whole Body Control pipeline on Krang, the low level controller follows the center of

mass trajectories planned by the high level controller, maintaining a stable balance while smoothly moving its joints [3]. The wheels of the platform can also be maintained at a steady position while the upper body moves. This creates an opportunity to automatically and quickly pick up objects with the arms. In addition, a sensor can be attached on the robot that is able to collect reliable transform data for localization and estimating the state of the robotic system, with a guarantee of reliability that the sensor will be maintained in a stable orientation and experience smooth transitions in its position.

1.1.3 Krang Capabilities

Out of this control scheme are borne various capabilities for Krang. Krang can pick up objects from tables while balancing on two wheels, due to the fact that end-effector position can be maintained. Due to its ability to locomote and still maintain the end effectors in a stable position, Krang can transport sensitive items across a space quickly without fear of dropping those items. Krang can pick up heavy objects by manipulating its weight torque, or use leverage to pry open a door with a stick. The gamut of experimental possibilities is extensive for this platform. However, it still lacks the ability to accomplish any of these tasks autonomously, or with even sparse human direction.

1.2 Research Objectives

Given that Krang is such a heavy and capable machine, its abilities are still limited without the advantage of autonomy. An ulterior motive was formulated to make the robot less dependent on human oversight to accomplish the objectives outlined above. As such, various sensors were explored to estimate and localize Krang in its environment. Further usage of a sensor system on Krang for a more advanced analysis and understanding of the environment was also desirable, in particular, identifying target objects to pick up or navigate to, or localizing in an environment and recognizing obstacles.

Ultimately, Krang would need to accomplish many of these tasks on its own. The

objective of this design project is to develop a base level of autonomy in order to complete the experiments without human presence. In addition, due to the advantageous positioning of the sensor system on the torso of Krang, it can be utilized to determine the dynamics of the mounting point, such that these dynamics are then used as a redundancy for checking that Krang is in a stable balancing position. For example, a very high acceleration at the top of the spine may not match the expected trajectory generated by the high-level controller. If such is the case, then it can be deduced that the spine is not behaving as it should, and the robot should default to some safe behavior.

1.2.1 End Effector Control

All seven joints on each of Krang's two serial manipulators can be controlled individually using a remote joystick. However, the task is slow and requires a complex combination of joystick buttons and toggles to move the end effector in discrete steps. It is simpler to put the end effectors in a pre-determined configuration and place objects in the robots grippers before doing any locomotion tasks. In order to allow Krang to pick up objects on its own, it is necessary to develop an inverse kinematics formulation and control method for the manipulators, such that the end-effector can be placed at a desired position and orientation with respect to a target object. The inverse kinematics formulation can inform joint-level PID controllers, or command torques directly to move the hardware.

1.2.2 Visual-Servo Control

The desired pose of the end effector is determined by the pose of a target object in front of the robot. This requires a robust method of target object feature tracking in the image for 3D position estimation, which informs the desired end-effector pose [4] [5]. The camera will analyze the scene from the torso bracket that forms the base of both manipulators, in an eye-to-hand configuration for visual-servoing purposes. The location of the end effectors is described with respect to the camera frame using forward kinematics. The attributes of the

target object are known and utilized to detect its location with respect to the camera frame. The method of estimating the 3D location of a target object and the end effector, and using that information to inform the control of a grasping manipulator is called a position-based visual servo (PBVS) scheme, or a 3-D localization problem [4].

1.2.3 Base Angle Approximation

Approximating the base angle of Krang in order to achieve stabilization redundancy or an alternative method of balancing was an intriguing prospective for this project. Krang has one inertial measurement unit in the base of the robot, which publishes very noisy IMU data. Methods of filtering this data result in a delay of smoothed IMU data, which is a potential point of failure in the critical balancing objective. In addition, since the high level controller is receding horizon based MPC, having a delayed estimation of the dynamics can be a cause for undesired control trajectory generation. However, by attaching the camera to a relatively stable joint on the robot from which reliable transform velocity and acceleration data can be interpreted, and then applying the correct transformations to the base, the angle of the base and its rate of change can be obtained.

1.2.4 Localization

The above capabilities require Krang to have some level of autonomous locomotion in order to round out a fully autonomous system. In particular, with the ability to sense its environment, it was naturally necessary to be able to localize Krang automatically at initialization. By mapping the lab environment and recognizing its position, Krang can be instructed to find and complete tasks along familiar paths. [6] shows that odometric measurement for differential drive robots can incur constant errors over time that would cause the inferred position of the robot to drift over time. In addition, Krang sometimes exhibits quick changes in pitch, causing a propagation of effects on the body, which ultimately changes its exact wheel position on the ground. By including a method of correcting this

drift through camera data, a more accurate estimate of Krangs position over the course of experiments can be ensured.

CHAPTER 2
REVIEW OF PREVIOUS WORK

2.1 Visual Servo

In [4], a general vision-based control scheme is defined as,

$$e(t) = s(m(t), a) - s^* \tag{2.1}$$

where $m(t)$ describes a set of image measurements, and a describes potential additional knowledge about the system, like camera parameters. These two parameters are used to describe the current value of features in the image, s. s^* describes the desired value of features in the image, and so their difference informs the controller about what actions need to taken to reduce the error $e(t)$ in image space. This formulation encompasses many of the approaches in visual-servo control. A method for Cartesian position-based visual servoing is presented in [7] for a camera mounted directly on the end effector. In this work, the target object frame is described with respect to the camera frame by mapping the *j*th feature point of the object to the image plane using photogrametric equations. The target object features are assumed known from a reference CAD model. After estimating the pose of the object with respect to the end effector, a desired end effector pose is defined. A control formulation for the manipulator is described to command changes in joint positions, which minimize error to the desired end effector pose. A similar method is utilized for the work in [8], however, the pose of the target object is estimated severally on image features, utilizing different image filtering and analysis techniques for detection of position and orientation with a reference CAD model. All of these techniques follow the general form described in Equation 2.1.

Most of the previous works separate pose estimators from controller design. The focus

of most of these works is in improving estimation of the target object pose from visual data, rather than designing control algorithms for minimizing $e(t)$. To that end, existing manipulators are used for the grasping portion of the project without undertaking rigorous stability measures. However, in [9], the controller adopts two novel adaptive estimators to estimate the positions of both the end effector and target object online from visual feedback of a stereo camera in an eye-to-hand configuration. It is shown that estimation errors for the pose of the target object and end effector converge to zero over time, enabling accurate pick-up of the target object with a guarantee of stability.

For the implementation of a visual servoing algorithm on a mobile robot, [10] has utilized SIFT and CAMSHIFT algorithms for object recognition for target object pose estimation. In addition, a stable balancing control is realized by estimating center of gravity through forward kinematics of the manipulator while conducting grasping operations. The robot uses a stereo camera to recognize the center point of a target object for each camera, and then estimates the distance of the object from the camera using a disparity calculation. A hybrid image-based and position-based visual servoing system is utilized for visual servoing.

2.2 Localization and Base Angle Approximation

Much work has been done in literature to estimate the world transform of mobile platforms using depth and visual sensors. An accurate example, [10] used stereo vision to track the three dimensional position and orientation of a mobile robot, showing a range of error of 20mm to 60 mm. And [11] shows that a visual-servoing robot can track objects to be grasped while simultaneously predicting the robots position and orientation. SLAM algorithms have been a useful tool for robotics applications. In particular, [12] used a drone-mounted stereo camera to map a large outdoor environment to a point-cloud mesh. Structure from Motion (SfM) and Multiple View Stereo (MVS) algorithms were utilized to accurately reconstruct 3D structures in the scene. The experiment was conducted in

an outdoor environment for mapping a building facade, and deviation was less than 6 cm. Similarly, [13] utilized a depth sensor to build a point cloud representation of stairways and railings, and localized the robot relative to the stairs to enable stair-climbing. The ZED camera used in that experiment localized to within 10 centimeters for distances up to 4 meters. This point tracking system allows the camera to track its pose as well as to localize. A wheel inverted pendulum is controlled in [14] with a vision-based adaptive controller that balances the robot on two wheels and follows a human target. A 2-DOF inverted pendulum is stabilized in [15] using visual estimation of the inverted pendulums top coordinates in an inner loop, while the position of the base is obtained from position encoders. For the application in this paper, the base angle of the robot is the angle parameter used to build a simplified model of the inverted pendulum.

CHAPTER 3
TECHNICAL APPROACH

This chapter justifies what tools and methods will be utilized to accomplish the needs outlined in Chapter 1 for this robot. Various technologies are considered for the selection of an environment-sensing sensor. Examples of measuring the environment include understanding depth of obstacles and recognizing objects and their state around the robot. For example, a robot may need to understand the location of a door on a wall, how far it is, and whether the state of the door is ”closed” or ”open”. These measurement capabilities also make it possible to localize and estimate the state of the sensor. Different serial manipulator control methods are also discussed.

3.1 Selection of Sensor

Some potential technologies to consider for depth sensing were LIDAR, ultrasonic and vision systems that return depth information. All three technologies were considered to meet localization and base angle approximation objectives, with varying degrees of accuracy. However, since Krang is meant to interact with objects and carry out tasks in a known environment, being able to recognize specific objects and their dimensions was ideal. [16] showed that a digital representation of an object can be constructed from point cloud depth data, which means that potential familiar shapes like trays, cups or other objects Krang is meant to interact with, can be recognized using ultrasonic and LIDAR technologies. While inexpensive, ultrasonic sensors suffer from loss of accuracy in the presence of soft obstacles (padding on our laboratory walls), and generally exhibit more noise in depth readings, and less range than LIDAR. They also offer much less resolution than needed for such a project. While LIDAR systems are extremely detailed and accurate, [17] showed that multi-view stereo vision systems proved capable enough to meet a LIDAR-based ground

truth model of an outdoor scene with sufficient accurary for most applications. In addition, vision systems leave open the possibility of leveraging the many advanced vision techniques in future work, which was a major motivation for selecting vision cameras as the technology of choice.

3.1.1 RGB-D vs Stereo

For selection of a depth-sensing vision system, two common technologies were considered. RGB-D sensors utilize an infrared sensor next to the visual camera in order to measure depth for every pixel in the image. This is accomplished by projecting an array of infrared points on the scene and results in a very accurate reading of depth, given that other sources of infrared light do not interrupt the reading. Stereo cameras utilize two cameras configured at a fixed distance apart facing the same scene. If various intrinsic parameters for the two individual cameras are known, along with the extrinsic parameters describing their transformations relative to each other, common descriptors read in the pair of images provided by the cameras can be utilized to measure the distance of the descriptors from the camera [18]. Stereo cameras also provide depth data on a pixel-by-pixel level.

Two brands were considered for the two technologies. The Microsoft Kinect 2.0 is a common RGB-D camera used for this application, while the StereoLabs Zed was a promising stereo camera option. [19] did a study of the performance of each camera and found that the Kinect was slightly more accurate in its readings. However, the ZED offers built-in technologies relevant to our desired application and the ability to access these features out of the box. In addition, stereo camera technology, particularly on the ZED, exhibited very good performance in outdoor environments, which is a glaring limitation for RGB-D cameras. For these reasons, the StereoLabs ZED was chosen for this project.

3.1.2 ZED Camera Features

The ZED Stereo Camera possesses two lens that are spaced 120mm apart (Figure 3.1). Out of the box, the camera API provides 6-DOF positional tracking data. In addition, the API also allows for spatial mapping, which can stored as a pre-defined map of the experimental environment. The camera clips objects that are closer than about three-quarters of a meter, resulting in no depth data for objects in that area. This is fine for our localization or base angle approximation, since the robot arms may at times occlude the cameras view within the 0.70 meter clipping distance, and the arms don't need to be tracked themselves. Camera resolution and frame rate, coordinate settings, and various other parameters can be set up at the initialization of the camera. The settings for the experiments conducted are tabulated in Table 3.1. By using a WVGA format, a higher frame rate is utilized to ensure more reliable tracking data.

Figure 3.1: ZED Stereo Camera (Source: StereoLabs)

The camera is attached to Krang with a rigid mount that is angled by 65 degrees, or one that is at 45 degrees (Figure 3.2). These two orientations offered the best point of view depending on the experiment being run, and selection of the orientation was specified in a configuration file.

Table 3.1: Parameters Set on ZED Camera

Parameter	Setting
Resolution	768x480
Frame Rate	100
Coordinate System	Z-up
Coordinate Units	Centimeters

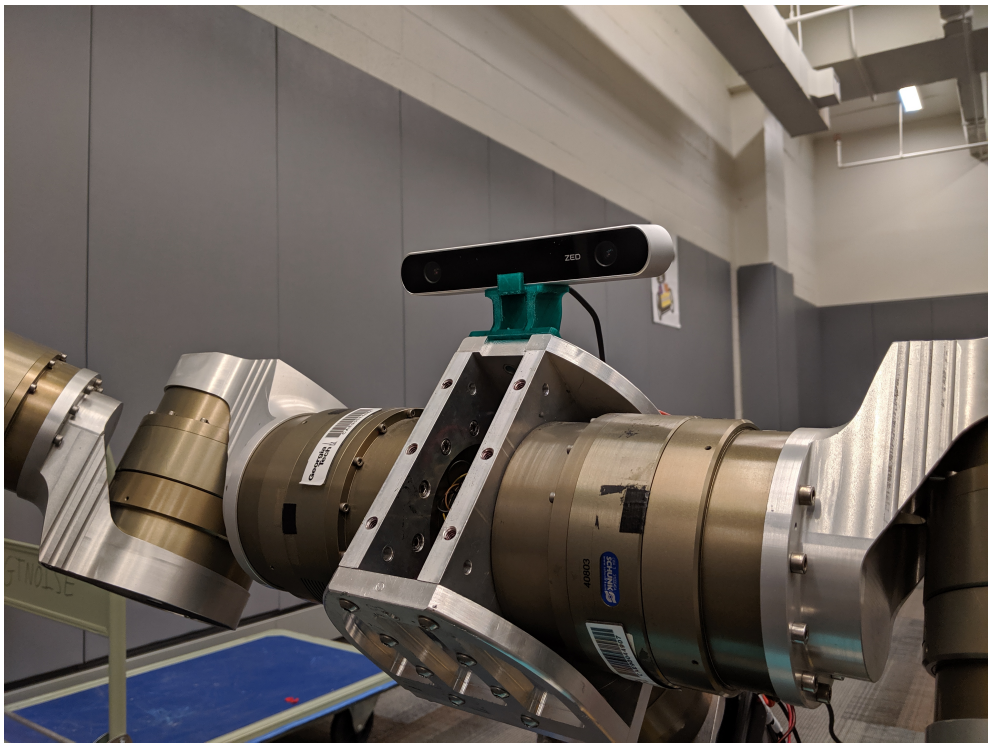

Figure 3.2: ZED Camera on 45° Mount

3.2 Positional Tracking

The shoulder bracket of Krang offers a good perspective of the field in front of Krang. The torso joint that the brackets attach to is also a relatively stable joint of the robot. In addition, this is a good perspective between the two arms of Krang, ensuring that the arms - or whatever they're carrying - don't occlude the view of the camera due to the low carrying position, and also offering a view of where the end-effectors are if the arms are angled up,

or the waist angled down. Measurements are taken from this point.

The measurements provide the transform of the sensor on the top of the shoulder bracket, with respect to a world origin defined at camera initialization, or with reference to a pre-defined world origin that the camera localizes to. Krang dynamics are computed in the Dynamic Animation and Robotics Toolkit (DART) [20]. This toolkit provides a variety of tools, including the ability to transform between sections of the robot. Through DART, a series of transformations applied to the camera provides transform data of the base. The torso and waist joints between the base and camera are actuated by an interprocess communication (IPC) framework utilizing ACH [21] channels to transmit state and control signals. DART maintains a simulated replica of the hardware orientation by reading from these ACH channels on the robot. The transformations are updated in realtime and the proper base transform approximation is obtained in DART.

3.3 End-Effector Control

The arms of Krang are articulated structures consisting of seven revolute joints. A tree-structure can be utilized to describe the position of the end effector with respect to the base torso bracket. This is denoted as,

$$O_{EE} = f(\Theta) \tag{3.1}$$

where Θ is an $nx1$ vector containing the joint angles of the n degrees of freedom of the manipulator. In particular, Θ describes the $(\theta_1, ...\theta_7)$ joint angle values, and O_{EE} describes the $(x, y, z, \theta, \phi, \psi)$ of the end effector.

There are two more coordinate origins to consider here. The origins of the camera, and target object are O_C and O_O respectively (Figure 3.3). A transformation from the camera coordinate system to the end effector is represented by ${}^{C}T_{EE}$, where,

$$^{C}T_{EE} = {}^{C}T_{Base}\ {}^{Base}T_{EE}$$

$^{C}T_{Base}$ depends on the mounting fixture used to attach the camera to the base of the arm, but does not change when the arm is moved, whereas $^{Base}T_{EE}$ is a function of Θ. It is not shown explicitly in Figure 3.3, however the path can be traced from O_{Base} to O_{EE} through the arm. $^{O}T_{C}$ is directly measured by the vision system. Then $^{O}T_{EE}$ can be described by,

$$^{O}T_{EE} = {}^{O}T_{C}\ {}^{C}T_{EE}$$

Substituting above equation for $^{C}T_{EE}$

$$^{O}T_{EE} = {}^{O}T_{C}\ {}^{C}T_{Base}\ {}^{Base}T_{EE}$$

$^{O}T_{EE}$ is the transformation between the frame of the end effector, O_{EE}, and the target object O_O and is shown as an orange dashed line in Figure 3.3. The goal is to drive the error between the two to zero.

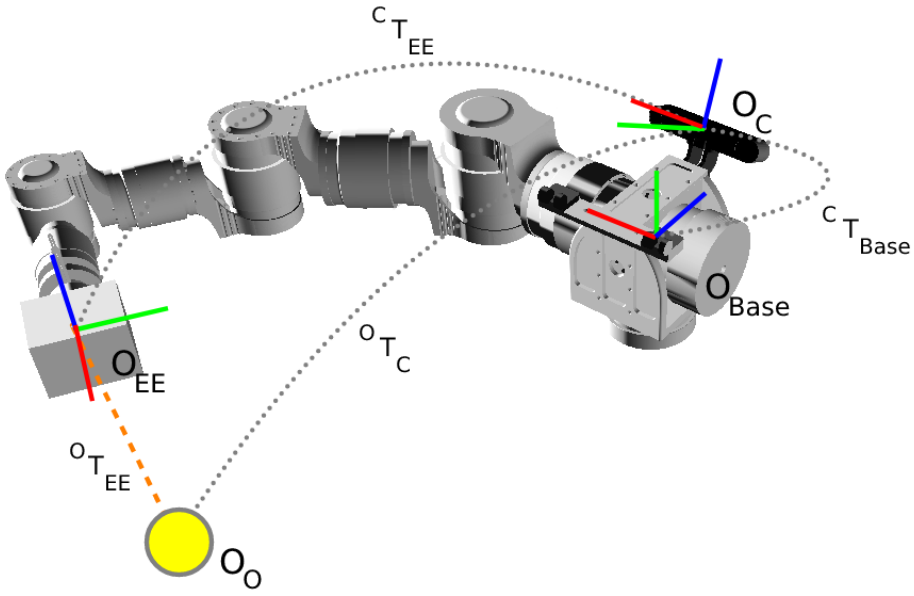

Figure 3.3: Transformations for a Single Arm of Krang

3.3.1 Inverse Kinematics Formulation

Since we want to minimize the error between the end effector and target object in Cartesian space, we must solve for the set of joint angles Θ that will give us the desired end effector pose described by the formula,

$$(\Theta) = f^{-1}O_{EE} \tag{3.2}$$

Various methods exist for solving the inverse kinematics of an articulated structure. Algebraic methods exist for a restricted class of structures, and the number of nonlinear equations to be solved rises exponentially with the number of DOFs [22]. Expressing the equations for our 7-DOF system is non-trivial. Rather, iterative methods were considered for solving the inverse kinematics problem. Of these, a common method is Jacobian Inversion, in which the Jacobian J relating the Cartesian space of the end effector X to joint space angles Θ as expressed in Equation 3.3, is inverted and multiplied across X to relate differential changes in Cartesian space to differential changes in joint space. However, this method suffers from slow computation for large DOF systems due to the inverse matrix computation time, and if the Jacobian is non-square and consequently non-invertible, a pseudo-inverse of the Jacobian must be computed which introduces some numerical error [22]. An optimization-based method was utilized by looking at the above equation as a minimization problem as expressed in Equation 3.4, where the cost function is of the form $\frac{1}{2}||PX - b||^2$ [23]. This solution to the inverse kinematics problem gives fast and accurate results. The details of implementation are explained in Chapter 5.

$$X = J(\Theta)\Theta \tag{3.3}$$

$$\begin{aligned} &\min_X \frac{1}{2}X^T G X + g^T X \\ &G = P^T P, g = -P^T b \end{aligned} \tag{3.4}$$

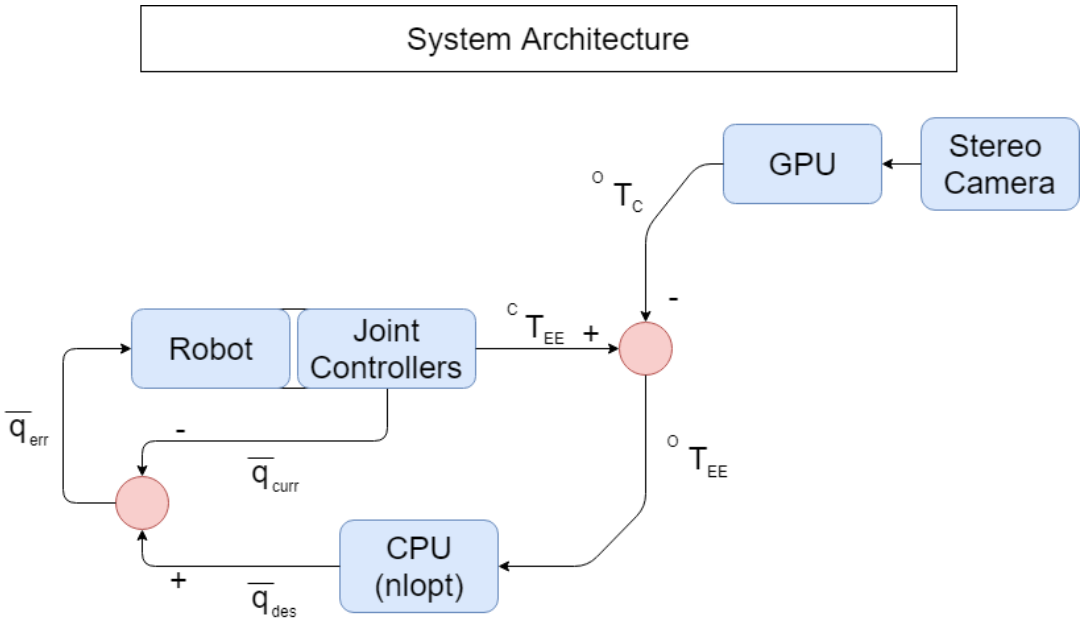

Figure 3.4: System Architecture

CHAPTER 4
OBJECT DETECTION AND LOCALIZATION

As mentioned in the introduction, a key experiment to verify whole body control is working is to carry a tray of water across a space while balancing on two wheels. In order to complete the tray carrying experiment, ideally Krang would initialize, find and pick up the tray and then take it to a new location. In current experiments, Krang is initialized, put into a standing position with plenty of clear space ahead, and given the tray and object to hold. It needs a space of about 5 meters to move forward and stop.

As one can surmise, this experiment requires heavy human oversight to ensure that Krang is operating the experiment in a safe space, while Krang is completely blind to the environment. With a short movement of 5 meters, there is virtually no drift in position estimation from odometry. However, by moving the robot back and forth for multiple passes of the experiment, some significant drift begins to occur.

4.1 Obtaining and Filtering the Target Object Position

The process described here utilizes the cameras intrinsic and extrinsic parameters, represented by a in Equation 2.1, to estimate the 3-D location of the tennis ball with respect to the camera, $m(t)$.

4.1.1 Detecting the Object

OpenCV [24] was utilized to detect a tennis ball (Figure 4.1), due to its bright color and simple shape. For this purpose, the raw image from the left camera was converted to an HSV map for analysis (Figure 4.2a). The HSV model represents color using three components. The hue component encodes the color information for each pixel in a range of 0 to 360, while the saturation component encodes the intensity of the pixel color, and the

Figure 4.1: Raw Image with Detected Region Wrapped in a Contour

value component specifies the brightness of the pixel color in ranges of 0 to 100. A specific advantage of utilizing an HSV map in our application is that the value component can be adjusted to fit a wide range of brightness of the scene being analyzed, while the hue and saturation terms remain in a tighter range. Thus, an HSV map is more robust to tracking the same object despite varying brightness in the scene being analyzed. A bounded range for each component was defined (Table **??**). Each pixel in the image has a 3-dimensional HSV value, and because only the pixels representing the tennis ball fall within the 3-dimensional defined range, they form the positive values in a binary mask (Figure 4.2b). The mask represents the region where the tennis ball was detected. A dilution filter is used to fill small holes that appear in the mask where the image pixels fell *outside* the defined HSV range, and an erosion filter is used to dissolve spots that appear outside of the mask where image pixels were detected *within* the defined range.

In order to obtain the position estimate of a point for grasping, a single point must be defined in image space. This is accomplished by fitting a contour around the mask detected in the previous step, which appears as a red line around the ball in Figure 4.1. The contour

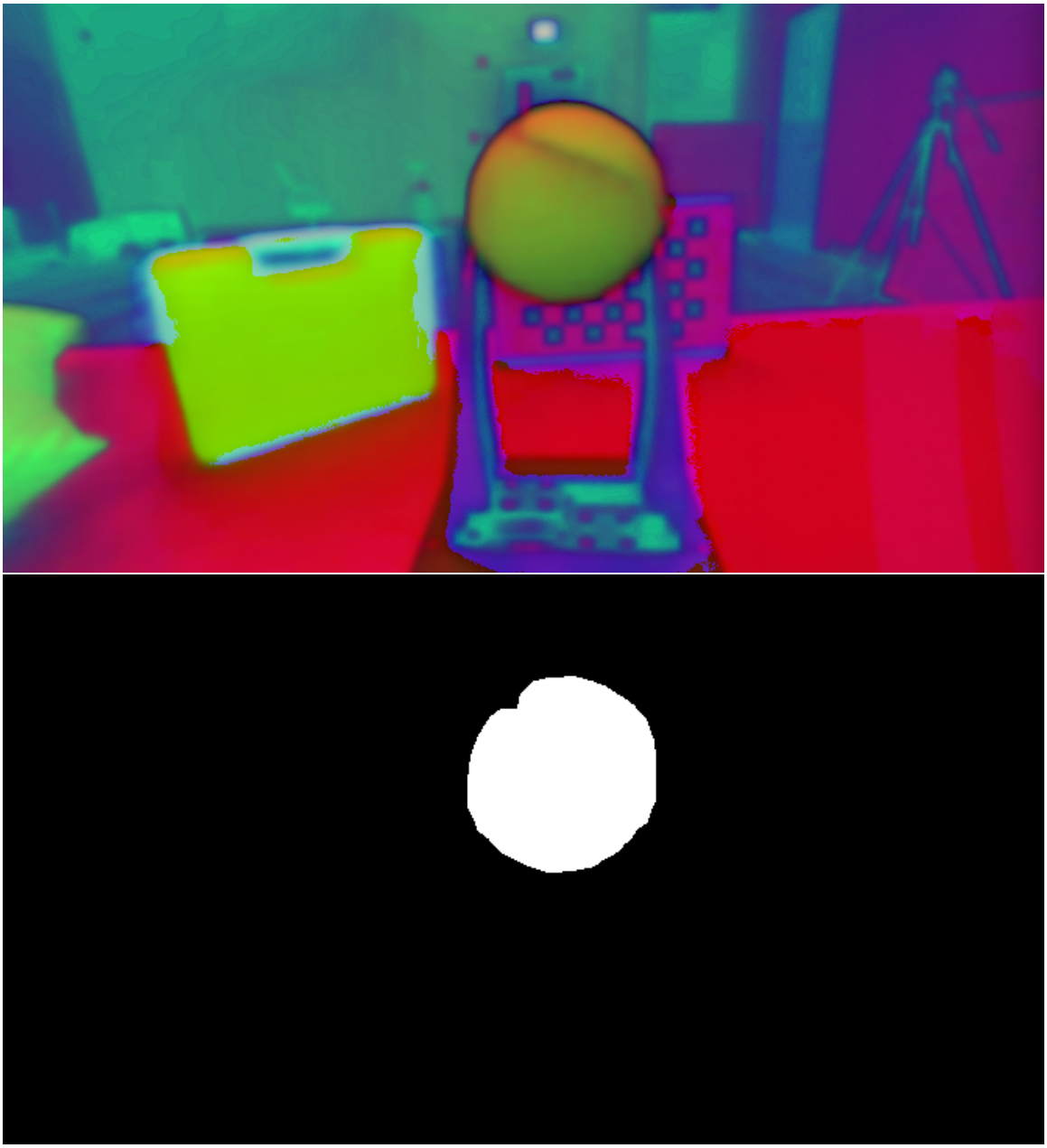

Figure 4.2: HSV Map of the Raw Image (top) Mask of the Detected Region (bottom)

Table 4.1: Range of HSV Values

Component	Value Range
Hue	29:64
Saturation	86:255
Value	6:255

is a curve joining all continuous points around a boundary, and is represented by an array of pixel values in the X and Y dimensions of the image space. Due to the erosion and dilation filters applied in the mask detection step, all random noise in the image is dissolved, leaving a mask of a single region, and subsequently, a single contour. Then, by taking an average of the X and Y array of values defining the set of pixels forming the contour, a good estimate of the center of the tennis ball is obtained as an (X_{avg}, Y_{avg}) coordinate point. This point appears as a blue dot near the center of the tennis ball in Figure 4.1. After tuning the HSV range and erosion and dilution parameters, a best fit for the tennis ball still left part of the ball undetected in some cases, particularly in low-light situations where part of the ball was shaded. However, the variation in the point (X_{avg}, Y_{avg}) only shifted by $\sim$1 cm.

4.1.2 Obtaining the Object Position

After finding the contour center in the previous step, the ZED API was utilized to obtain a depth map of the scene (Figure 4.3). Key descriptors are compact and distinctive regions of pixels in an image that are easy to track across multiple frames. The depth map is obtained by tracking key descriptors in the image and matching these descriptors across the two views from the stereo camera. Epipolar geometry is utilized to estimate the distance of the point from one camera sensor utilizing the intrinsic parameters of each camera and the extrinsic geometry between the two cameras. The ZED API maintains a point cloud representation of each descriptor detected in the scene, resulting in a 3-Dimensional cloud representation of objects in front of the camera. A provided two-dimensional pixel value in image space returns a three-dimensional position value from the generated point cloud through the ZED API, for any pixel location in the image. Thus, after obtaining the contour center and passing it to the API function, a three-dimensional position is recovered for the tennis ball and image updates are acquired from the GPU through a TCP connection every 10 ms.

Figure 4.3: Raw Image (top) and Corresponding Depth Map (bottom)

The 3-Dimensional position value of the tennis ball, extracted using this method, varies around the true value of the 3-Dimensional position relative to the camera. Figure 4.4 shows the true static position of the tennis ball as a green line, and variation around it over time. A running average value is taken for the positional data, giving a smooth trajectory very quickly for the most recent six data points, also shown in Figure 4.4. The variation from the true value decreases by utilizing the running average data. Table 4.2 shows the mean and true values for all three dimensions, as well as the variation. Figure 4.5 shows the same data with a moving tennis ball.

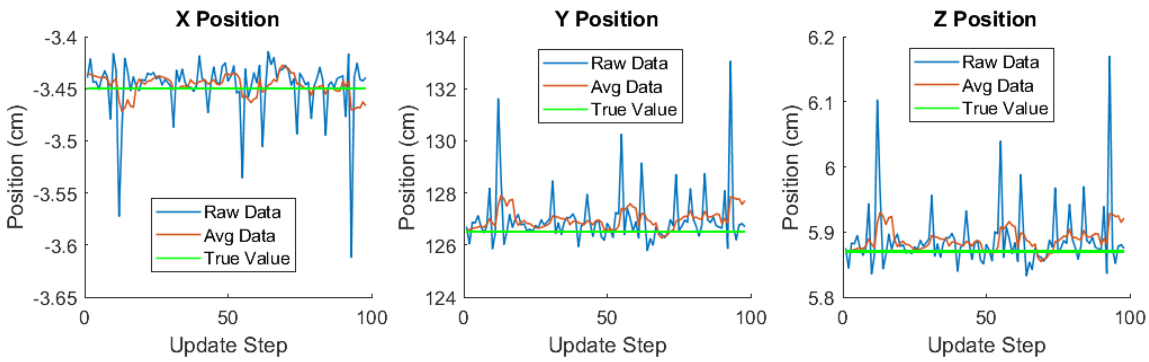

Figure 4.4: Position of Static Tennis Ball (100 steps = 7.5 seconds)

Axis	Mean μ	Variation σ	True Value
X	-3.4471	0.0285	-5.8
Y	126.9919	1.0500	124.5
Z	5.8884	0.0487	5.87

Table 4.2: Mean, Variation and True Value of Detected Positions (cm)

The tracking process suffers from some shortfalls due to the distinct vision and positional tracking tasks here. While color tracking is robust even with occlusion of parts of the ball due to a single image being analyzed from the left camera, the same occlusion may result in a loss of tracking ability due to occlusion of the calculated contour center from the right camera. In other words, only one camera is needed for detecting the ball, but both cameras are needed for estimating 3D position of the ball, and occlusion of one results in a failed depth reading. In addition, large expanses of low contrast regions behind the ball may also result in a failure to detect the three dimensional position from image space inside this low contrast region. The nan values returned from the ZED API in these cases is rejected and the last known position is held until a new reading is acquired. At times when the camera is able to track the ball however, the depth value has never wavered far from the true value, probably due to robust tracking and filtering software on the ZED.

While an advanced vision application was not worked on here, the set of tools available show that it would be quick to implement an object detection algorithm for any target

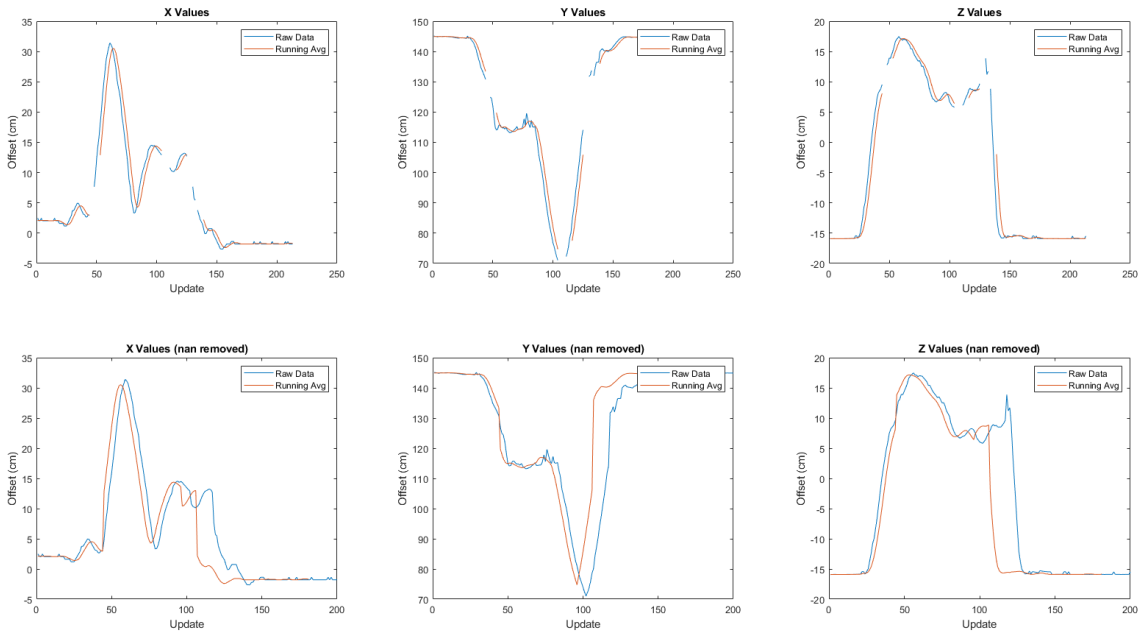

Figure 4.5: Positional Data Over Time with Running Average Smoothing (100 steps $\approx$ 7.5 seconds)

object. In particular, the YOLO library [25] can be readily utilized to train a convolutional neural network with any desired dataset, and recognizes objects much faster than traditional vision detection pipelines that track key descriptors, convolutional features or localizers in feature space [25]. The source software for YOLO is freely available. With the added benefit of having access to the stereo camera API, a dense point cloud reconstruction of any particular target object is quickly obtainable, enabling recognition of the object from its point cloud geometry, as [26], [27] and various others have shown methods for accomplishing this. In Krang's case, objects of interest may be trays, cups, and other objects utilized to fulfill Krang's capabilities. Finally, by analyzing the features of a known model, the orientation of the target object can be estimated as well.

4.1.3 Workspace for Visual Servoing

For this project, the workspace for arm manipulation is restricted to the cone of view in front of the camera, and so the union of the viewing angle of the stereo camera with the

workspace of the arm within that cone will form the workspace for this application. With a viewing angle of 90° horizontally and 60° vertically, the union of the stereo camera views is shown in Figure 4.6. The cones are clipped at the minimum distance for gleaning depth information. The workspace of the manipulator intersects with most of this region, and clips it at the fully extended singularity configuration of the arm, closing the union of the camera and arm. If the target object moves out of range, the arm will reach as far as it can without exhibiting any instability or loss of objective to reach the goal.

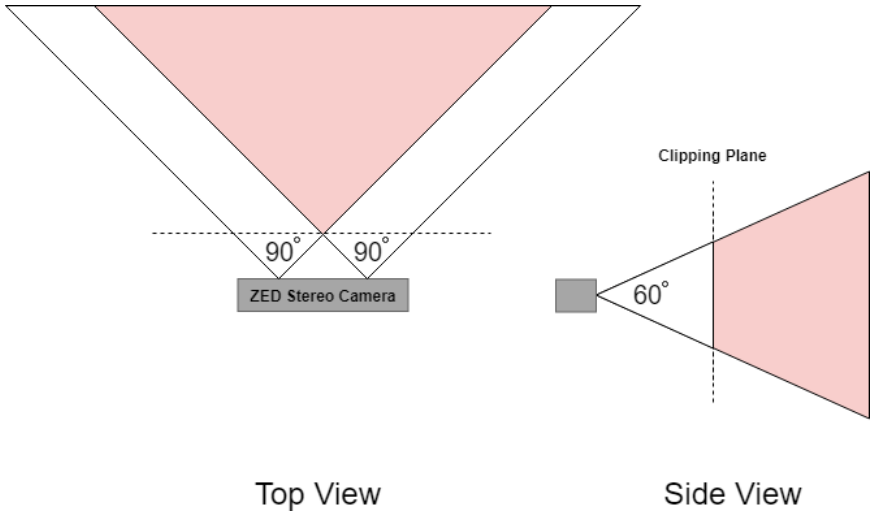

Figure 4.6: Viewing Space of the Camera

CHAPTER 5

END-EFFECTOR CONTROL

In order to move the end effector to a desired position, a minimization problem is formed for determining optimal velocity targets for the joints of the arms. Computation of the inverse kinematics is spread out over many time steps, making the control of the arm very reactive to changes in position or orientation of the target pose. This chapter goes into detail about how the problem is formulated, and shows results and details for simulation and hardware experiments.

5.1 Optimization-Based Inverse Kinematics Formulation

Once the location of the target object with respect to the camera frame is found, the difference in location between the end effector and target object in Cartesian space is directly known. The optimization algorithm minimizes this difference by formulating the problem as a quadratic programming problem with the general form as described in [23],

$$\min_X \frac{1}{2} X^T G X + g^T X \tag{5.1}$$

where the cost function is of the form,

$$\frac{1}{2}||PX - b||^2 \tag{5.2}$$

and $G = P^T P, g = -P^T b$.

The cost function is a weighted sum of quadratic terms penalizing X for deviating from some desired X^*. In this case, X is the velocity of the end effector, penalized for deviation from reference transformational and rotational velocities. P and b arrays express multiple

tasks, with weights ($w_0 ... w_n$) describing relative importance of each task.

$$P^T = [w_0 P_0 \quad w_1 P_1 ... w_n P_n]$$
$$b^T = [w_0 b_0 \quad w_1 b_1 ... w_n b_n]$$

For moving the end effector to a desired gripping pose, three tasks were specified, including position and orientation error minimization and speed regulation for both of Krangs arms, bringing the number of tasks to be solved to six. In particular, for one arm we need to formulate the P and b vectors for each task as,

$$\begin{aligned} P^T &= [W] \cdot [P_{Pos} \;\; P_{Or} \;\; P_{Reg}] \\ b^T &= [W] \cdot [b_{Pos} \;\; b_{Or} \;\; b_{Reg}] \\ [W] &= [w_{Pos} \;\; w_{Or} \;\; [w_{Reg}]^{7x1}]^T \end{aligned} \tag{5.3}$$

The form of the cost function for velocity error between the end effector and target object is,

$$\frac{1}{2}||\dot{x} - \dot{x}^*|| \tag{5.4}$$

where $\dot{x} = J(q)\dot{q}$ and $\dot{x}^* = Kp_{Pos}(x_O - x_{EE}) + \dot{x}_d^*$.

Since there is no goal velocity, $\dot{x}_d^* = 0$. Then, equating Equation 5.4 to Equation 5.2 reveals that,

$$\begin{aligned} P_{Pos} &= J_v \\ b_{Pos} &= -Kp_{Pos}(x_O - x_{EE}) \end{aligned} \tag{5.5}$$

where J_v is the derivative of the Jacobian of the manipulator.

Since there is only an (x, y, z) coordinate detected by the camera for the tennis ball, a desired orientation $(\theta_d, \phi_d, \psi_d)$ is formulated for the arm to move to, resulting in the pose $(x, y, z, \theta_d, \phi_d, \psi_d)$. The desired orientation is defined to be parallel to the cameras orientation, but can be set to any arbitrary orientation - i.e. if the orientation of the target object is detected as well. With a parallel orientation, the end effector does not occlude the target object during the grasping operation. The error between the end effector orientation and the desired orientation of the target object is to be calculated. DART utilizes an angle axis representation for rotations. Obtaining the rotation component of the transform of the end effector in DART returns an angle axis definition of a quaternion, represented as $Q = (w, x_q, y_q, z_q) = (cos(\frac{\theta}{2}), Vsin(\frac{\theta}{2}))$. That is, given an angle axis, the subsequent quaternion is formed via the set of equations,

$$V = (v_x, v_y, v_z)$$
$$x_q = v_x sin(\tfrac{\theta}{2})$$
$$y_q = v_y sin(\tfrac{\theta}{2})$$
$$z_q = v_z sin(\tfrac{\theta}{2})$$
$$w = cos(\tfrac{\theta}{2})$$

The angle axis values returned by DART are utilized to form a quaternion vector for the target object in simulation using these equations. Given that Q_{EE} is the quaternion of the end effector, Q_{DO} is the quaternion of desired orientation, Q_w describes the length and Q_{xyz} describes the unit vector of the quaternion, the error is represented by,

$$Q_{err} = Q_{DO,w} * Q_{EE,xyz} - Q_{EE,w} * Q_{DO,xyz} + Q_{DO,xyz} \times Q_{EE,xyz} \tag{5.6}$$

If the dot product of the two quaternions is negative, then Q_{DO} is negated to prevent the arm from trying to swing around the long way to follow a changing orientation, which otherwise results in a self-collision. Let ω be the angular velocity of the end effector, $\dot{q}$ be the angular velocities of all joints, J_w be the angular velocity Jacobian of the end effector,

and $\dot{J}_w$ be the derivative of angular velocity Jacobian of the end effector. Then,

$$\dot{w}_{ref} = -Kp_{Or} * Q_{err} - Kv_{Or}\omega$$

is used in,

$$\begin{aligned} P_{Or} &= J_w \\ b_{Or} &= -(\dot{J}_w\dot{q} - \dot{w}_{ref}) \end{aligned} \tag{5.7}$$

Finally, for speed regulation,

$$\begin{aligned} P_{Reg} &= I^{7x7} \\ b_{Reg} &= -Kv_{Reg}\dot{q} \end{aligned} \tag{5.8}$$

where $\dot{q}$ is set to zero in the b_{Reg} term in order to reduce velocity.

The P and b vectors for a single arm are then formed by substituting Equations 5.5, 5.7, and 5.8 into 5.3,

$$\begin{aligned} P^T &= [W][J_v \;\; J_w \;\; I^{7x7}] \\ b^T &= [W][Kp_{Pos}x_{ref} \;\; \dot{J}_w\dot{q} - \dot{w}_{ref} \;\; 0] \end{aligned} \tag{5.9}$$

The weight ([W]) and gain parameters are tabulated in Table 5.1.

Table 5.1: Parameters for IK Simulation

Parameter	Value
w_{Pos}	1
w_{Or}	1
w_{Reg}	[0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]
Kp_{Pos}	[100, 100, 100]
Kp_{Or}	[15, 15, 15]
Kv_{Or}	[0.2, 0.2, 0.2]
Kv_{Reg}	1
Kv_{Joint}	[50, 50, 50, 50, 50, 50, 1]

The NLOPT library [28] [29] was utilized to solve this QP problem at every update step, and returned a 7x1 array of velocity targets for the joints, $\dot{q}_{opt}$. The formulation was first deployed in simulation with a keyboard-controlled target pose and a single arm. Figure 5.1 shows the rise of error when the target pose is manually changed in the software, and the resulting activity in the seven joint positions. When the error is low, the joint positions remain constant and the velocity remains zero. When the error increases due to moving the target position or orientation, the joint positions and velocities begin to change as the arm moves the end effector to the new goal orientation. Figure 5.2 shows how the arm reacts to a large offset in initial error, ${}^{O}T_{EE}$. Damping and friction coefficients were added in simulation to produce a smoother response that better represents behavior of the hardware.

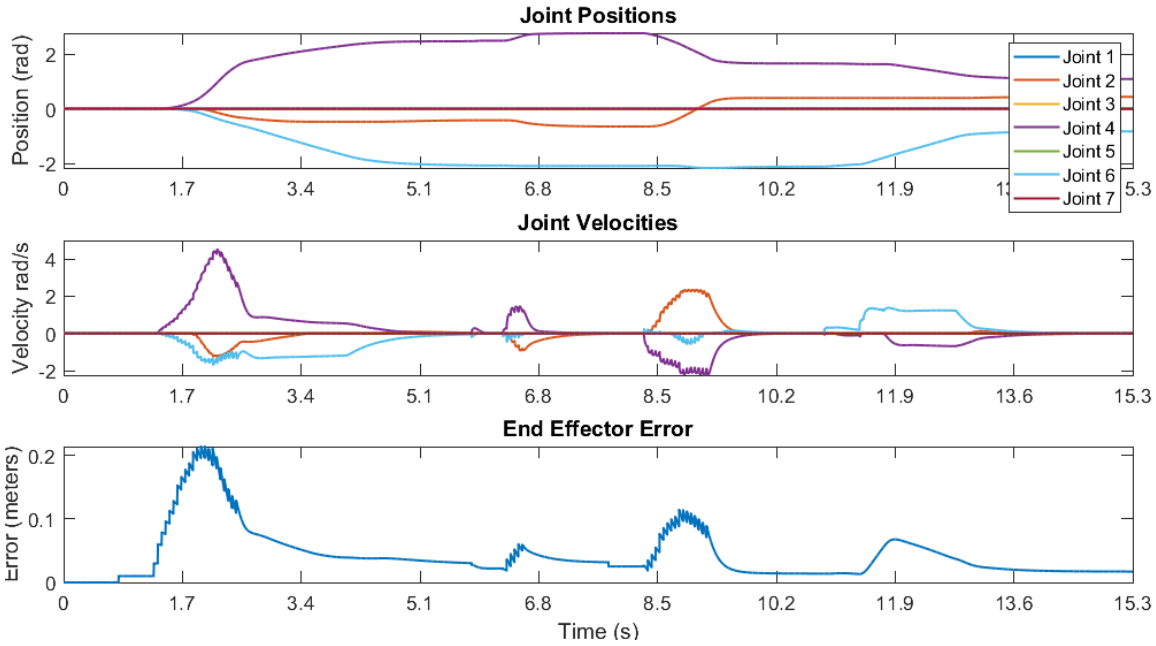

Figure 5.1: Joint Movement and Changing End Effector Error

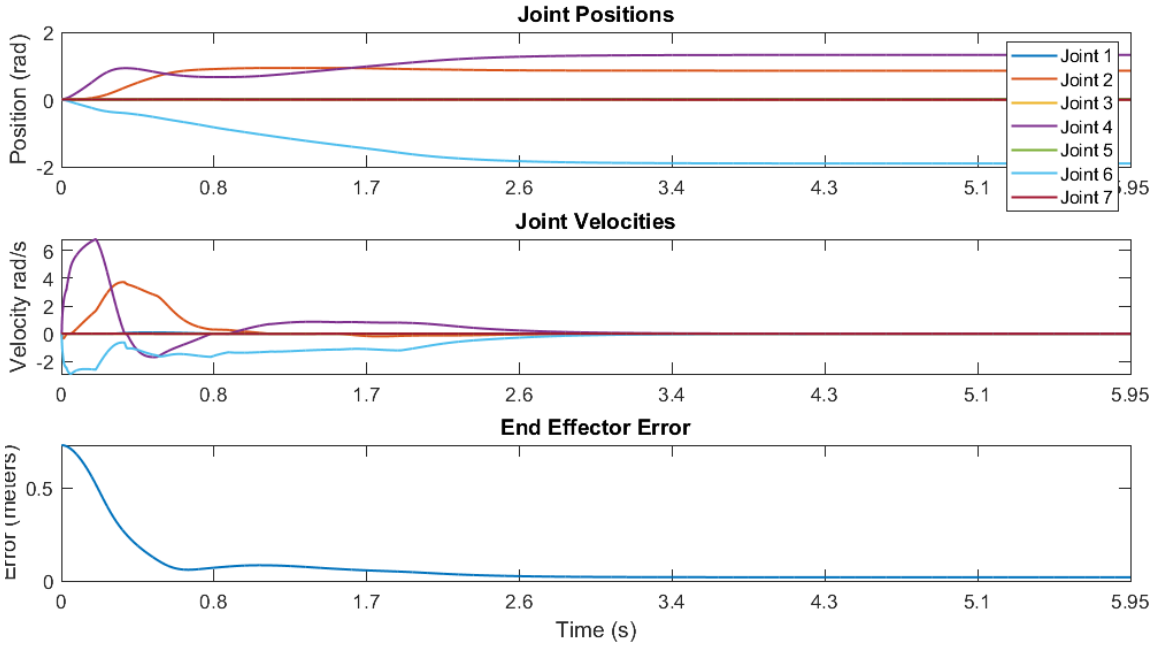

Figure 5.2: Joint Movement and Static Starting End Effector Error

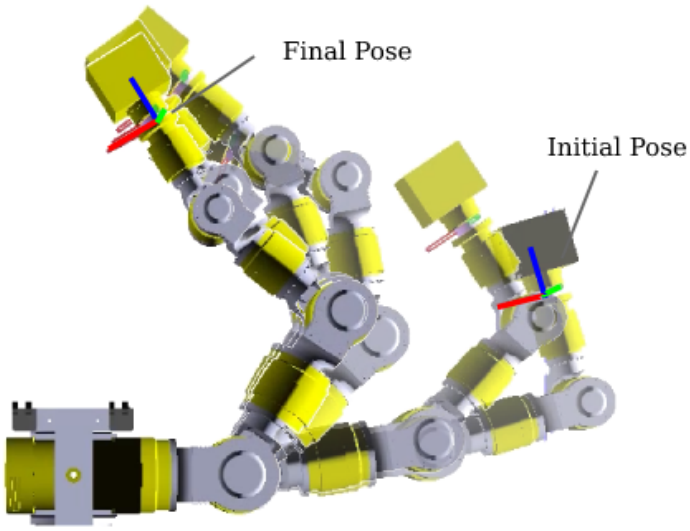

Figure 5.3: Visual of Movement for the Plot of Figure 5.2

The results were demonstrated to work for the solution of both velocity and acceleration targets in simulations in [30]. The goal in that work was to maintain an arm configuration, rather than moving the arm over a large arc to follow a target object. A potential issue for large movements is that the arm exhibits self-collision in an effort to meet the target position and orientation. This self-collision is avoidable on hardware due to joint position limits, however the optimization algorithm does not account for these limits and this can cause issues for solving IK for hardware. This issue was resolved in [23] by defining constraint equations as part of the optimization operation, but was not implemented in this project. In addition, for certain extreme configurations it is still possible that the tip of the manipulator can collide with other parts of the robot. Thus, there is a need for inclusion of a control formulation that avoids self-collision. However, if the arm has an initial orientation that is close to the grasping orientation with the end effector in front of the camera, then there is no risk of self-collision even with large position errors. The initial configuration in Figure 5.3 is an example of one such starting orientation, where the arm is extended out to the side, but the orientation of the end effector is within 90 degrees of the grasping orientation. For

this reason, a bounding box was defined as a safe working space for the arms, and repeated simulation tests within this bounding box showed that self-collision did not occur even for extreme errors in end effector location.

In simulation, a torque input is directly commanded for the manipulator,

$$u(t) = K_{vJoint}e(t) = K_{vJoint}(\dot{q}_{EE} - \dot{q}_{opt}) \tag{5.10}$$

For hardware, due to the highly nonlinear properties of the harmonic drive motors, as well as the unknown relationship between torque and current, the on-board PID controllers must be utilized. The output of the optimization algorithm, $\dot{q}_{opt}$, is directly sent as velocity targets to the joint-level controllers. In order to prevent the arm accelerating at a very high rate, a limit on joint velocities is defined at arm hardware initialization.

5.2 Hardware Control with Linear Interpolation

The manipulator hardware is composed of seven Schunk Universal Rotary Actuators of varying sizes, with built-in holding brakes that lock the motors in position while a control signal is not present. The actuators are controlled by internal logic, which receives parameters from the user including position, velocity, acceleration or current. For the case of inverse kinematics control, a velocity parameter was passed to each motor, and the internal PID control algorithm actuated the motors simultaneously to the desired velocity target.

For the first hardware deployment experiments, the simulated end effector target was initialized in DART at the pose of the real end effector. Then, the target was transformed in DART via manual keyboard commands to allow the arm to follow discrete translations of 1 cm or rotations of 3 degrees, deviating little from the real end effector pose for each update step. The inverse kinematics is solved just as it is for simulation. A plot of the hardware joint movements and error is shown in Figure 5.4. Although the velocity read from the joint hardware appears as a series of steps in Figure 5.4, there is very little oscillation observed

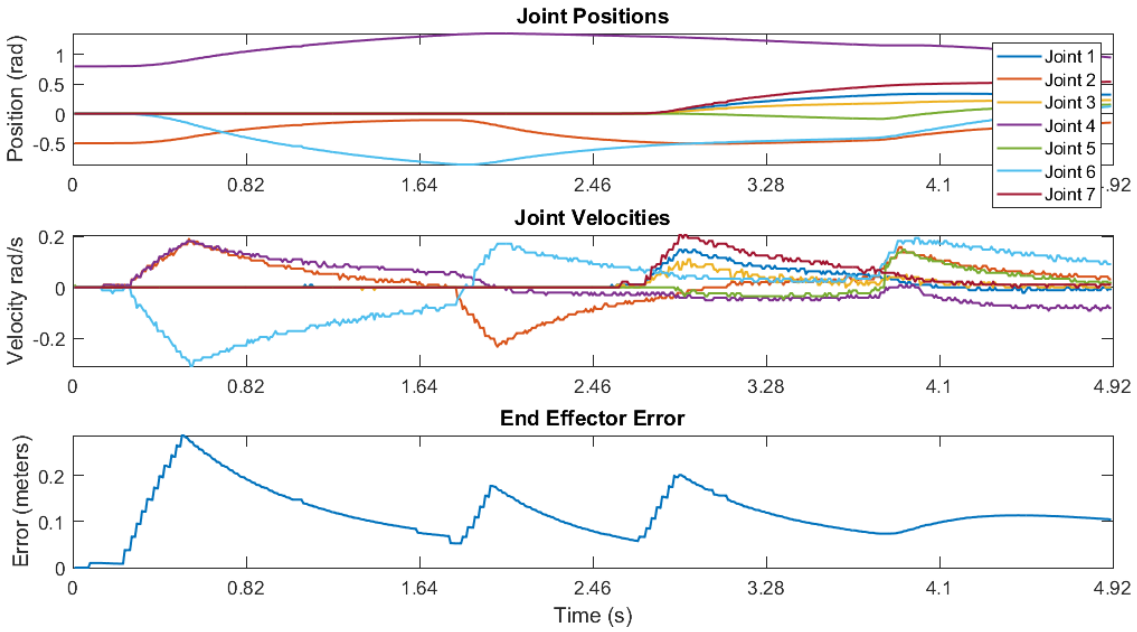

Figure 5.4: Plot of Hardware with Changing Error

in the velocity of the hardware due to the presence of high frictions and perhaps a robust internal PID algorithm.

This hardware experiment worked fine due to the small error introduced by moving or rotating the setpoint very little at each update step. However, by defining the setpoint to be at the location of the target object, the hardware may potentially start at an initial configuration far from the target grasping pose. There were two issues with directly sending velocity commands to the motors in this case. For one, the arm is capable of moving very fast, with a data-sheet specification of 25 deg/s. A large arc of movement can be dangerous for the safety of the robot or people around it, as well as inducing high momentum to potentially throw off the balance of the robot. Secondly, during hardware experiments on the arms, we found that the motors were prone to blowing fuses, requiring costly amounts of time for repair.

A quick fix for this issue is to limit the motors when they are saturated, by setting lower speed parameters at initialization. However, linear control theory cannot be applied to this system, resulting in more complex analyses of stability. Some general approaches to avoid-

ing saturation are integrator anti-windup, gain scheduling, and setpoint ramping/weighting [31]. Setpoint ramping was explored to gradually move the setpoint value utilizing a linear or first order differential ramp function, in order to send smaller setpoint increments to the joint-level controllers. This requires that the general PID control law is modified such that,

$$u(t) = K_p(e(t)) + K_i \int_0^t e(t')dt' + K_d \frac{de(t)}{dt}$$

becomes

$$e(t) = K_p(\beta r(t) - y(t)) + K_i \int_0^{\inf} r(\tau) - y(\tau)d\tau + K_d(\gamma \frac{dr(t)}{dt} - \frac{dy(t)}{dt})$$

where $r(t)$ is the setpoint and $y(t)$ is the process variable. β and γ in this case are referred to as the setpoint weights. β is typically set within a range of 0 to 1, with lower values giving a more sluggish response, while a γ value of 0 is preferred to avoid large transients in the control signal [32]. While the exact PID logic is internal to the joint level controllers, we have access to the process variable and setpoint values, giving us the error in the system. An intriguing prospect is to use this error as a parameter to set a variable β, and use the modified error $e_m(t)$ to calculate a new velocity target sent to the joint level controllers. The resulting formulation still results in an uneven distribution of velocity targets for the joints, sometimes failing to significantly reduce maximum speeds for some joints. A better alternative was to interpolate the error between the end effector and target object.

Let x_{EE} be the initial position of the end effector and x_O be the target position. Taking the difference between the two, $v = x_O - x_{EE}$, the target position is interpolated as,

$$x_{LERP} = x_{EE} + \alpha \frac{v}{||v||} \tag{5.11}$$

with α = 0.7. Then $\dot{dx}_{ref}$ in Equation 5.5 becomes $\dot{x}_{ref} = x_{EE} - x_{LERP}$.

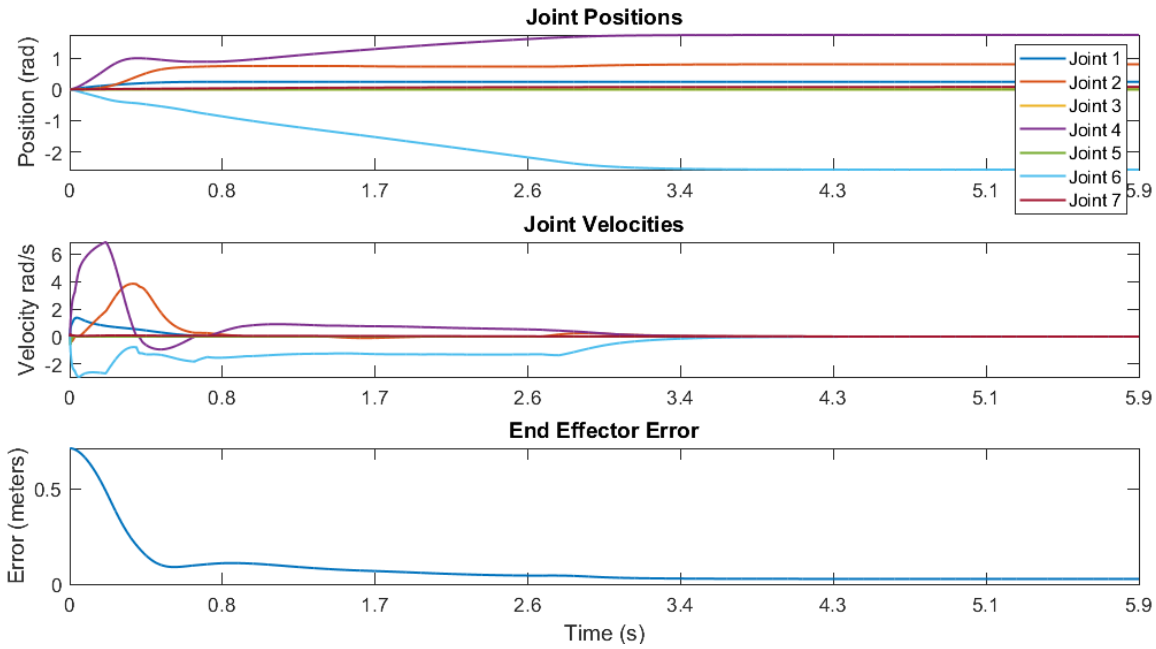

Figure 5.5: Initial Error with no Interpolation (Simulation)

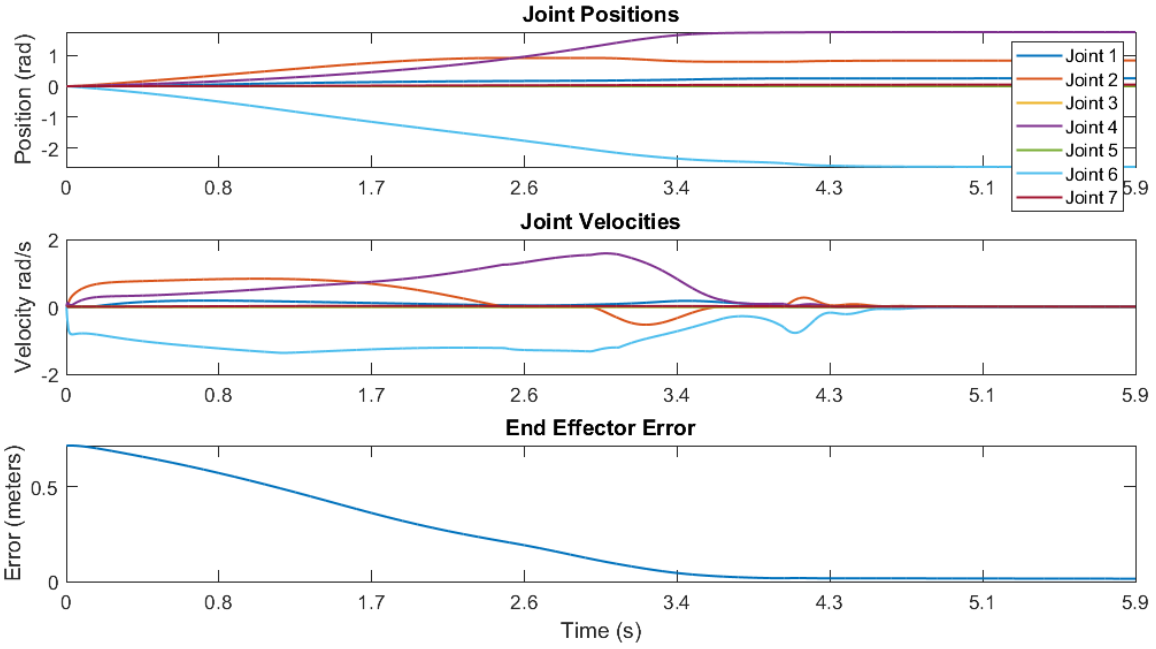

Figure 5.6: Initial Error with Interpolation (Simulation)

The arm moves into the new pose within the same amount of time, but the speeds of the arm are much lower and more gradual. In particular, Joint 4 in Figures 5.5 and 5.6 has its upper speed reduced from about 6.9 rad/s to 1.6 rad/s. The interpolation was deployed on

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
LERP	0.1845	0.8323	0	1.5847	0	-1.3680	0.0263
No LERP	1.3734	3.8480	0	6.8805	-0.5586	-2.9396	0.1556

Table 5.2: Differences in Maximum Joint Velocities (rad/s)

hardware and resulted in gradual speed changes for the arm after some tuning of the gain value, α.

CHAPTER 6

BASE ANGLE APPROXIMATION AND LOCALIZATION

The addition of a visual sensor on the robot creates the possibility of building a redundancy in stability objectives for the robot, such as in [33]. In that work, a low cost CCD camera is utilized to estimate the angle of the inverted pendulum on a moving cart by tracking the tip of the pendulum. In that implementation, the framerate of the camera and updates to the controller (25 fps and 40 ms) are both $\frac{1}{4}$ of the rates used in this project (100 fps and 10 ms updates). While Krang is not a simple inverted pendulum, the vision system on its torso can be utilized to predict the angle of the base in a similar fashion. The camera in this application is also usable for guiding the overall platform to the target object. For this, a simultaneous localization and mapping (SLAM) algorithm in the ZED camera API is utilized to build an area file of the map utilizing a heuristic method for quick mapping.

6.1 Generating Smoother IMU Data

A noisy IMU leads to a delay in control since the IMU data must be smoothed utilizing a Kalman Filter. It was explored whether camera data provides a sufficient transform to obtain some of the data that IMU gives (Figure A.1). More detailed and general plots of IMU data are shown in Appendix A.

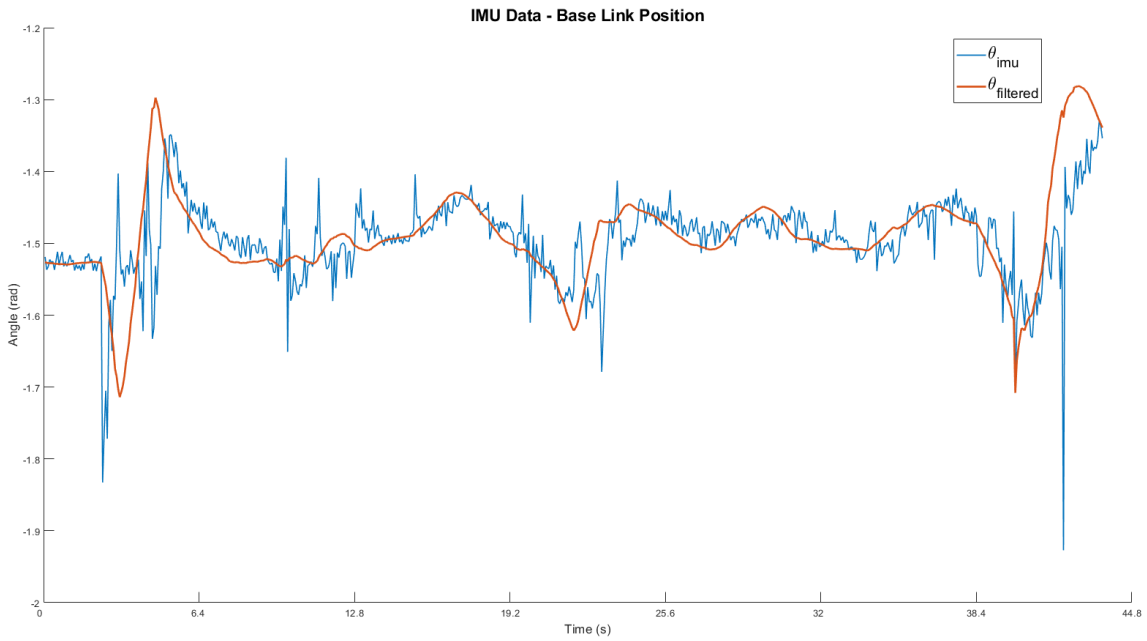

Figure 6.1: Plot of Raw IMU Data (blue) with Filtering (orange)

6.2 Mathematical Description of Transformations

Since DART uses a true-to-life model of Krang which has proven to work for hardware experiments, we can use DART to get an accurate transformation from the camera to base, ${}^{C}T_{B}$. Figure 6.2 shows the side view of the relevant bodies in the transformation, as well as the local coordinate systems on each joint. Since the camera is rigidly attached to the torso bracket, it can be assumed that the camera to bracket transformation remains constant. For subsequent transformations, DART mirrors the joint angles on the robot by reading from ACH channels at each update step, so the transformation to the base is derived at each update step. Equation 6.1 shows a mathematical representation of each necessary transformation.

A formulation obtaining the direct world transform of the base in DART, and the relative transform acquired through a series of transformations from the camera to base verified that they return the same values. Through the camera API, world transformations of the camera were acquired as 4x4 homogeneous transformation matrices describing the position

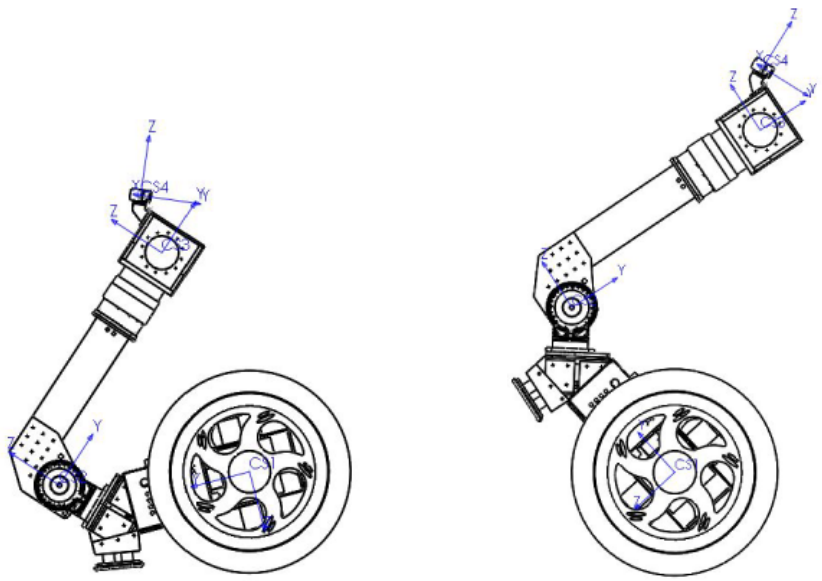

Figure 6.2: Krang Side View In Sitting and Standing Configurations (No Arms)

of the camera relative to a fixed world origin. This origin could be the transform of the camera at initialization, or the saved origin preserved in the area file for localization purposes. The ZED transform data is calculated for an origin at the back of the left lens. This data is transformed to the calibrated center of the camera body, matching the origin of the camera in our virtual model. DART transforms are applied at this origin. The DART isometries were converted to 4x4 homogeneous transformation matrices and multiplied with the correspondingly formatted ZED transform data in order to obtain the base configuration of the hardware (Equation 6.1).

The point where the camera is initialized becomes the global reference frame, referred to as the world frame. When Krang is initialized in a sitting pose, the starting angle of the camera is known, as well as its offset from an orientation parallel to the ground. In fact, it is simply the sitting angle of the base, q_{base}, summed with q_{waist}, and $q_{ZEDholder}$. Any subsequent movement, such as Krang standing up to balance on two wheels, is tracked by the camera. In addition, if the camera is continuously localizing, then it's angle relative to

the saved world origin in the area file is quickly obtained.

$$
\begin{aligned}
O_B &= O_C {}^{C}T_B \\
{}^{C}T_B &= {}^{C}T_T \, {}^{T}T_W \, {}^{W}T_B
\end{aligned}
\tag{6.1}
$$

where, given that ${}^{C}T_T$ is constant,

$$
{}^{C}T_T = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -0.398809 & -0.382941 & 0.083181 \\ 0 & 1.288468 & 0.178328 & -0.034185 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{T}T_W = \begin{bmatrix} cos(q_{torso}) & 0 & -sin(q_{torso}) & -0.028500 \\ 0 & 1 & 0 & -0.584 \\ sin(q_{torso}) & 0 & cos(q_{torso}) & 0.1088 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{W}T_B = \begin{bmatrix} 1 & 0 & 0 & -0.026 \\ 0 & cos(q_{waist}) & -sin(q_{waist}) & 0.387075 \\ 0 & sin(q_{waist}) & cos(q_{waist}) & -0.327803 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

6.3 Generating an Area File for the Experiment Environment

The camera is able to estimate its pose over time by tracking key static descriptors in the video feed it captures. By recording the spatial relationships between these descriptors, a spatial map of the environment itself is built. The result is stored as a heuristic map of the lab called an area file. The recording is done by toggling the spatial mapping feature in the ZED API and moving the camera around the environment through multiple passes to

generate an accurate map. The environment in the lab is generally static, although a new area file must be generated every time significant objects are moved. The world origin of the area file is defined to be the coordinate transformation at which the camera is initialized. An area file of the lab was generated using this process, and multiple initializations at different locations revealed that the camera was quick to localize its location in the lab. However, at times, active localization techniques were used such as changing the yaw of the camera or moving it forward and backward. The camera was accurate to within a few cm of the true world origin in the experiments. For the purposes of base angle approximation experiments, it was convenient to make the pose of the camera in Krang's sitting state the world origin.

6.4 Localizing

Passive localization acts as an observer along for the ride, always measuring the environment and updating Krang's location based on what it sees [34]. Passive localization works well enough that the localization task does not need to interrupt the experiments. However, localizing is necessary before starting the experiment. Since the ZED camera is capable of estimating distances up to 20 meters, a simple rotation in place is usually sufficient to localize the robot, and it can then move into position ready to start experiments. A default Boolean value is flagged to demonstrate when the camera has localized. Due to drift in the odometry readings over time, the localization task will continue to run passively, allowing the robot to conduct experiments, but cancelling drift by recognizing contextual information in the scene and closing the loop. A predefined (x, y) location for Krang to move to in order to find the objects to grasp is defined.

While autonomous movement is not programmed into the hardware, Krang can be moved using a remote joystick. After localization, the platform is moved to the goal position, and the camera is able to confirm when the robot is in position to start surveying the scene for objects to grasp.

CHAPTER 7

RESULTS

7.1 Visual Servoing

Figure 7.1 shows the position of the end effector on the hardware moving towards the 3-D location of the tennis ball as it is shifted around. The dashed line represents the X, Y and Z locations of the tennis ball over time as detected by the camera. The ball is not held still, although it is maintained in the same position for the last 6 seconds in the plot. The error is reduced to within a fraction of a centimeter.

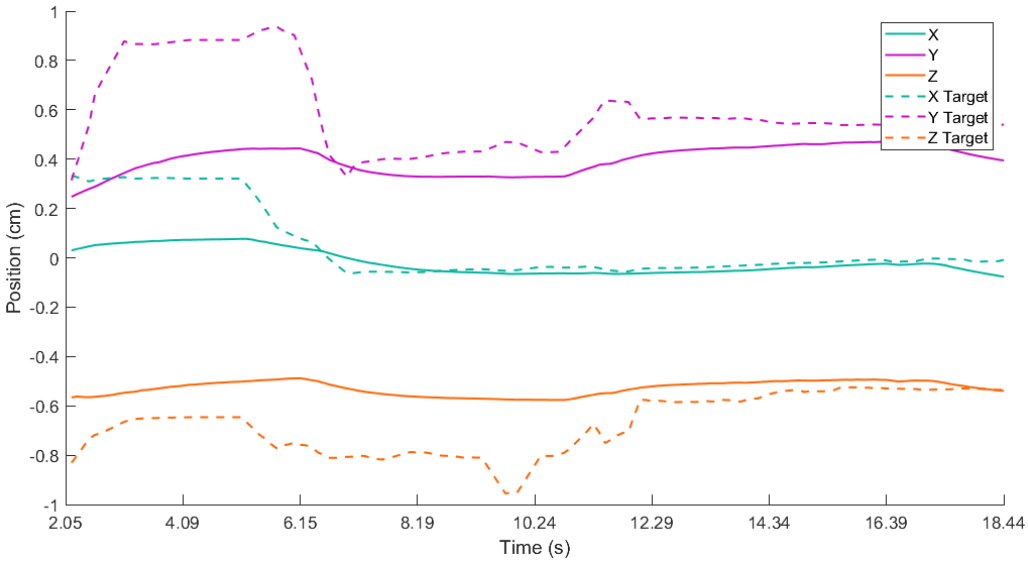

Figure 7.1: Plot of Position of End Effector and Target Object Over Time

Over multiple target locations, the actual error is within 2 centimeters along every axis. Because the arm does not change its orientation to capture the tennis ball, and because the grippers occlude the ball when they get close, the offset remains indefinitely for most goal positions. This can easily be resolved by moving the gripper to a location relative to the ball where occlusion does not occur, and then completing the gripping step by a standard and repeatable movement to move the grippers around the object. This was not done for this

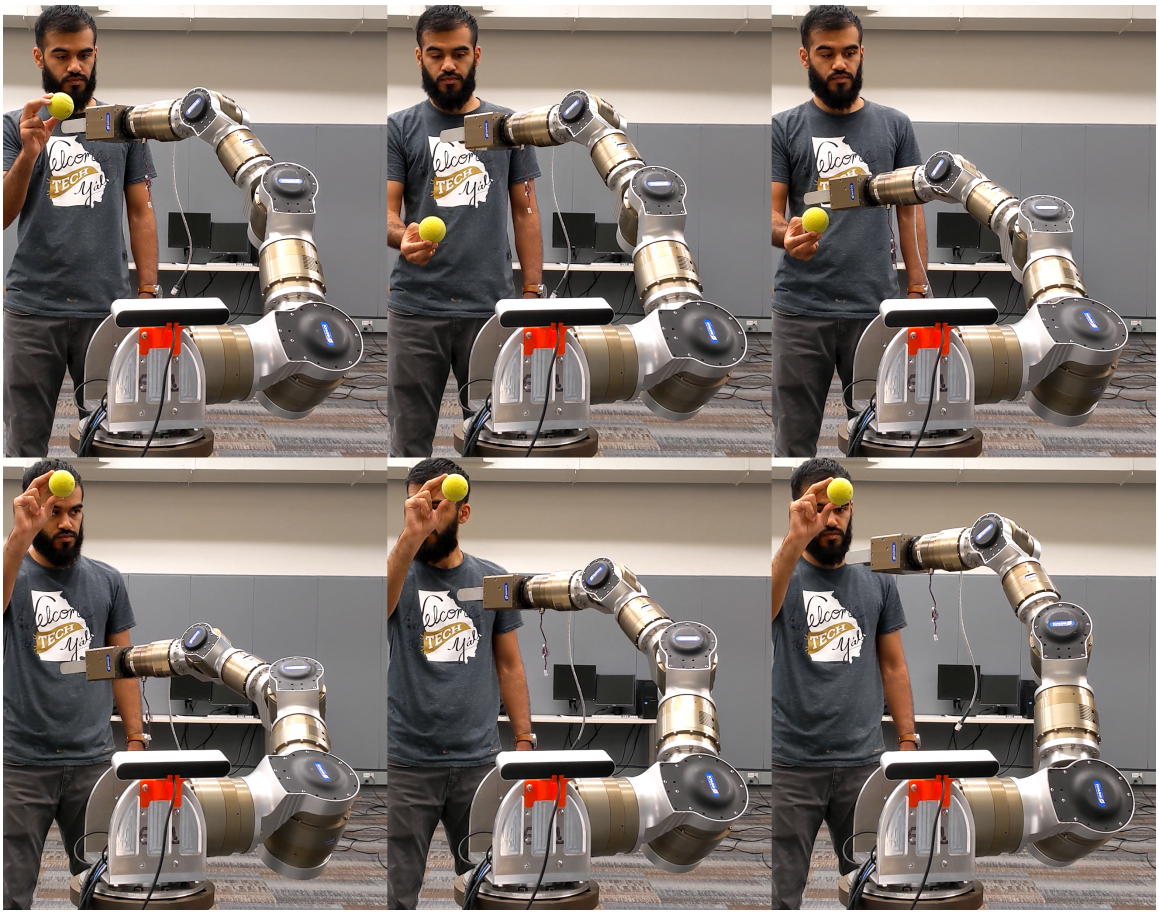

Figure 7.2: Arm Following a Moving Target Object

project because the gripper hardware was not functional, but it is easily applied. Figures 7.3 and 7.2 show the hardware moving towards the ball. The arm immediately adjusts it's trajectory in real time when the ball is moved. When the ball is out of reach, the arm reaches its workspace limit but exhibits no instability.

While the experiment was demonstrated on a single arm on the platform, the deployment to the full robot is quick because it has two arms that are exact replicas of the single one. Results of inverse kinematics on the full robot simulation were demonstrated in [30] for both arms.

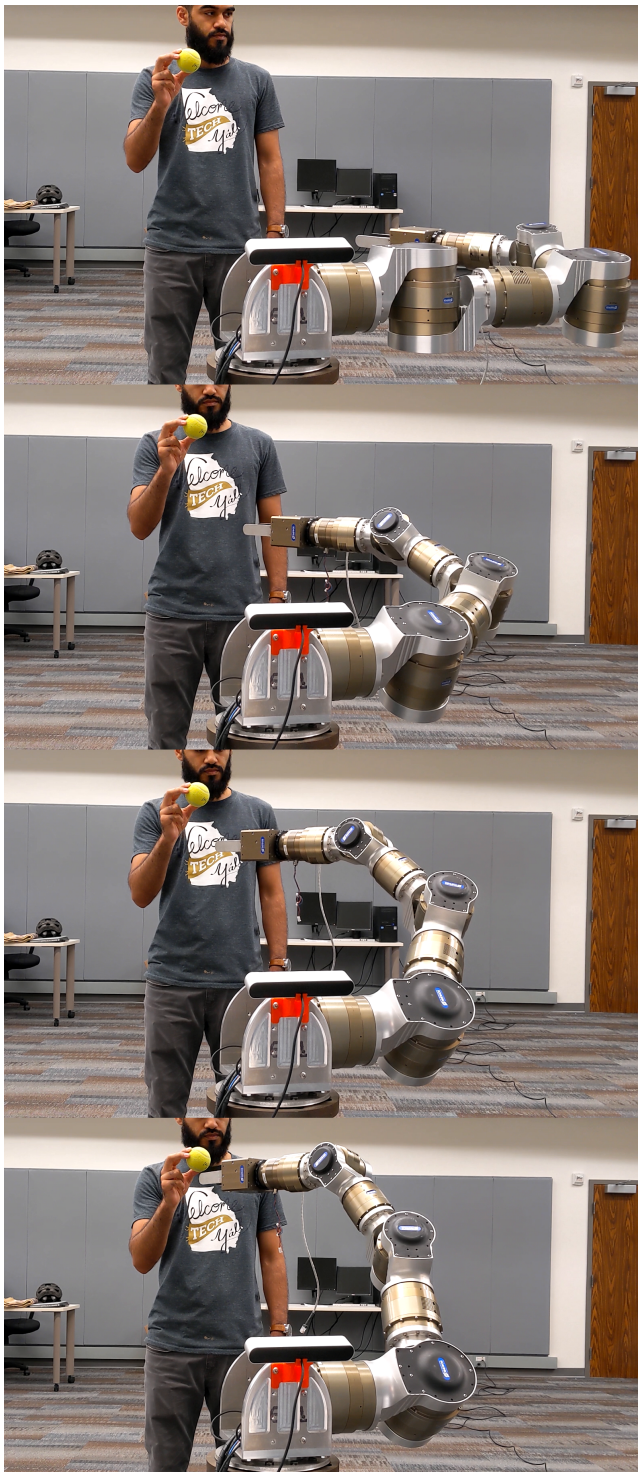

Figure 7.3: Arm Moving to Stationary Target from Resting Position

7.2 Base Angle Approximation

The positional tracking data from the camera matched very closely with the filtered data from the IMU, showing very promising results. It can be seen in Figure 7.4 that the camera data is following the trajectory of the IMU data. It is also clear that the IMU filtered data is shifted forward by a fraction of a second. This result shows that it is possible to assume the angle of the base with the camera data alone.

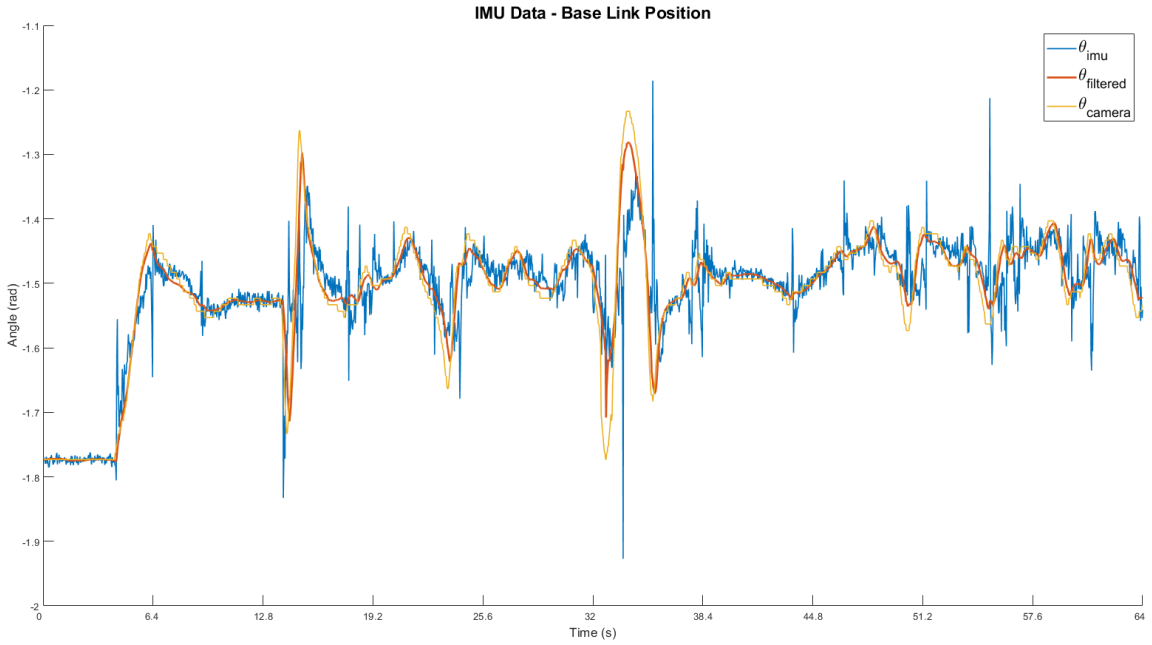

Figure 7.4: Base Link Position from IMU and Camera Data

This data will be useful for experimentation with the high-level controller. Since there is no ground truth data to compare to, the result of balancing with visual feedback must be compared to the result with feedback from IMU data. Since positional tracking was measured relative to ground truth measurements for the camera in Chapter 4, confidence coefficients for both camera data and IMU data can be calculated and utilized in a future sensor fusion project.

7.3 Localization

The robot was localized in the lab after generating an area file of the environment. Figure 7.5 shows the track line as the robot was moved. Points where the camera closed the loop are shown in the figure, and appear as step shifts in the track line. Localization did not occur immediately at camera initialization, but happened within a few seconds after shifting the robotic platform. Further experimentation must be done to compare the ground truth of robot position, estimate of position from odometry, and estimate of position from visual feedback. However, the camera has shown to be accurate to within centimeters even after being moved in a circle with a diameter of a few meters, as shown in the figure.

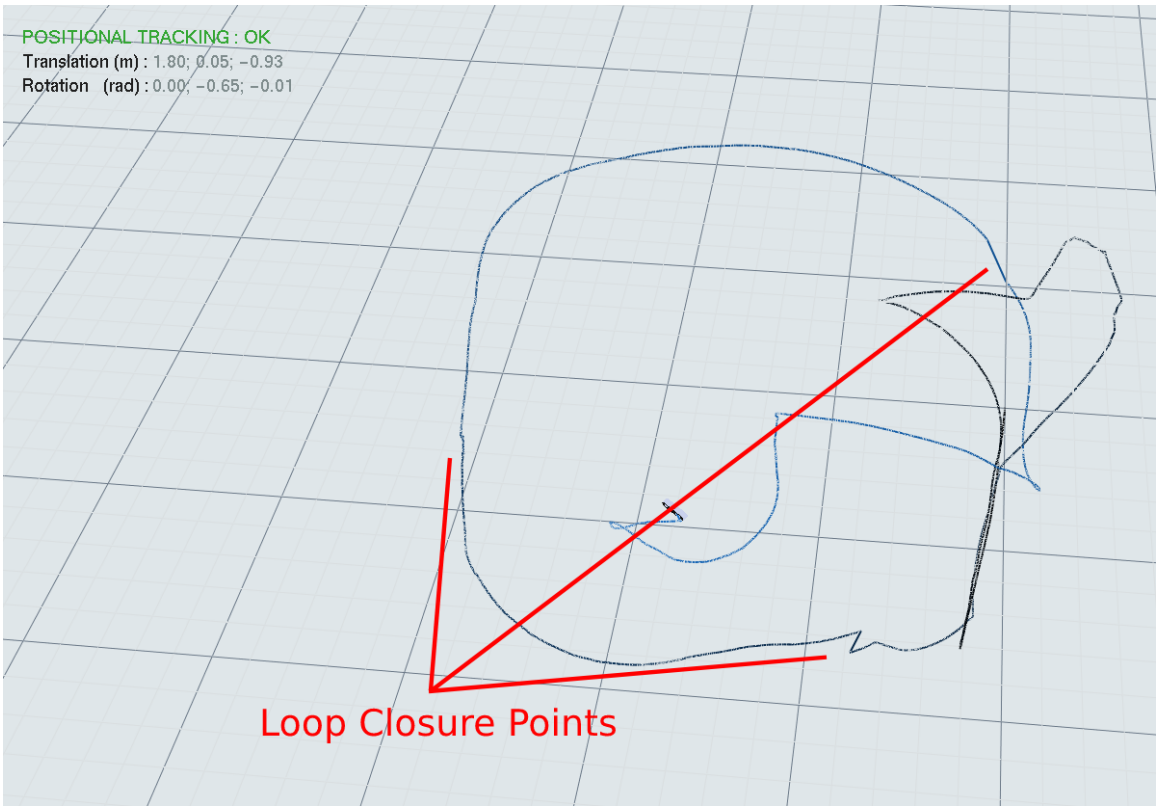

Figure 7.5: ZED Camera Tracking its Location around the Lab

CHAPTER 8
FUTURE WORK AND CONCLUSION

8.1 Future Work

8.1.1 Position-Based Visual Servoing

The current implementation works well for grasping target objects. However, it does so ”blindly”, since the camera does not detect the end effector. Rather, the CPU estimates where to place the end effector based on information it receives only about the target objects location. This is prone to produce error between the grippers and the target object in the case where there is bending in the arm, or the parameters of the manipulator, such as arm lengths, are not entirely accurate. A quick fix for this is to allow the camera to estimate the location of the end effector once it is near the target object, and to further drive the remaining error to zero before attempting to grasp.

8.1.2 Obstacle Avoidance/Safety Guarantees

Another shortcoming of the current implementation is that there is an assumption that there are no obstacles hindering the path of the end effector while it moves to grab the object. In addition, the locomoting robot cannot detect and plan around obstacles in order to get to a goal destination. The experiments performed in this work were done so in an uncluttered environment where it was safe to move the robot and arms. However, implementing some path planning into the solution would greatly enhance the robots autonomous capability, and safety during experiments.

8.1.3 Advanced Vision

Developing vision applications is a project that would need to be undertaken in order for the robot to pick up a multitude of objects, and to make intelligent decisions about order of pickup or which directions to move. As alluded to in the Introduction, Krang is a heavy machine whose intelligence would be greatly enhanced by understanding its environment. Implementing advanced vision that informs actions the robot takes is a key way to accomplish this.

8.2 Limitations

The implementation described here has some limitations. As described in Future Work, path planning should be implemented to ensure safe operation of robot locomotion or arm movement in the presence of obstacles. In addition, self-collision avoidance that the robot takes into account when optimizing for the target objectives should be applied. If occlusion of the target object occurs, the controller will remain at the previously detected target position, continuing to occlude the object. This might be desired if the arm has grasped the object and verifies that the grasping objective has been completed, but results in the arm getting stuck without completing the objective otherwise.

The target-sensing vision application was useful as a first implementation, but will not generalize to multiple targets or oddly shaped targets. For example, having contours around multiple detected objects will result in a target position in empty space between the objects where the contours average to. With a single object in the scene, an odd shape such as a hoop may result in a contour average in empty space at the center of the shape. An object larger than the gripper may need to be grabbed at an edge rather than the center. For these reasons, this is a special-case solution that only works for small, uniform objects that stand out from the rest of the scene in the image.

8.3 Conclusion

In this thesis project, a camera was attached to a wheeled, inverted pendulum humanoid in order to grant it the ability to autonomously locate and pick up objects using visual servoing techniques. In addition, the inclusion of the positional tracking capabilities of the camera allowed the robot to estimate the state of its base in order to provide redundant feedback for the high level controller, as well as to localize itself in a known environment.

For visual servoing purposes, a pre-defined target object was identified in the image, and its position in 3-Dimensional space was estimated. A running average calculation guaranteed that outliers in the readings or quick perturbations of the target object did not deviate the estimated target location drastically, and worked well to pass over erroneous readings. Forward kinematics provided the 3-Dimensional pose of the end effector and the error between the end effector and the target object was calculated. A quadratic cost optimization algorithm was utilized to solve the inverse kinematics of the 7-DOF arms in real time, with the expressed goal of minimizing the end effector error. The solution worked well to bring the gripper into position to grasp the ball.

Because the ZED stereo camera has a minimum clipping distance of 0.7 meters for measuring depth, and its smaller counterpart, the ZED Mini, has a clipping distance of 0.3 meters, the latter option would be a better stereo camera for visual servoing on this robot. The shortfall of the ZED Mini is that it has a shorter overall depth reading ability, which may affect localization efforts. The ZED Mini also possesses a built-in IMU, which may help for base angle approximation. Implementation of the ZED API does not change between the two cameras, so it would be quick to install the ZED Mini.

Localization of the platform was accomplished by recorded an area map of the experimental space, and localizing at initialization of the camera. Locomotion of the platform to a pre-defined target location was accomplished with a remote joystick, and the robot was able to recognize when it was in position to conduct experiments such as pick up of objects.

Localization ran passively in the background while the camera simultaneously tracked its pose over time. The pose was used to estimate the angle of the base, providing better estimates than the noisy IMU sensor in the base. This resulted in a better estimation of COM of the robot, which is critical for higher level objectives and safe, balanced operation.

Appendices

APPENDIX A

IMU PLOTS

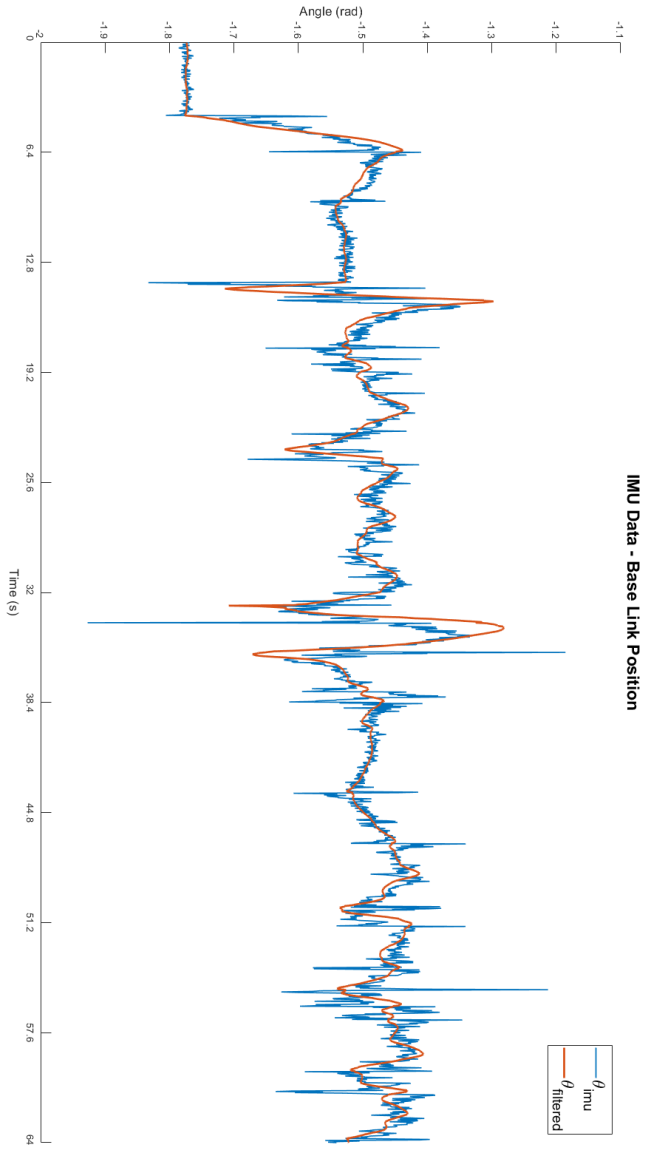

Figure A.1: High Range IMU Data

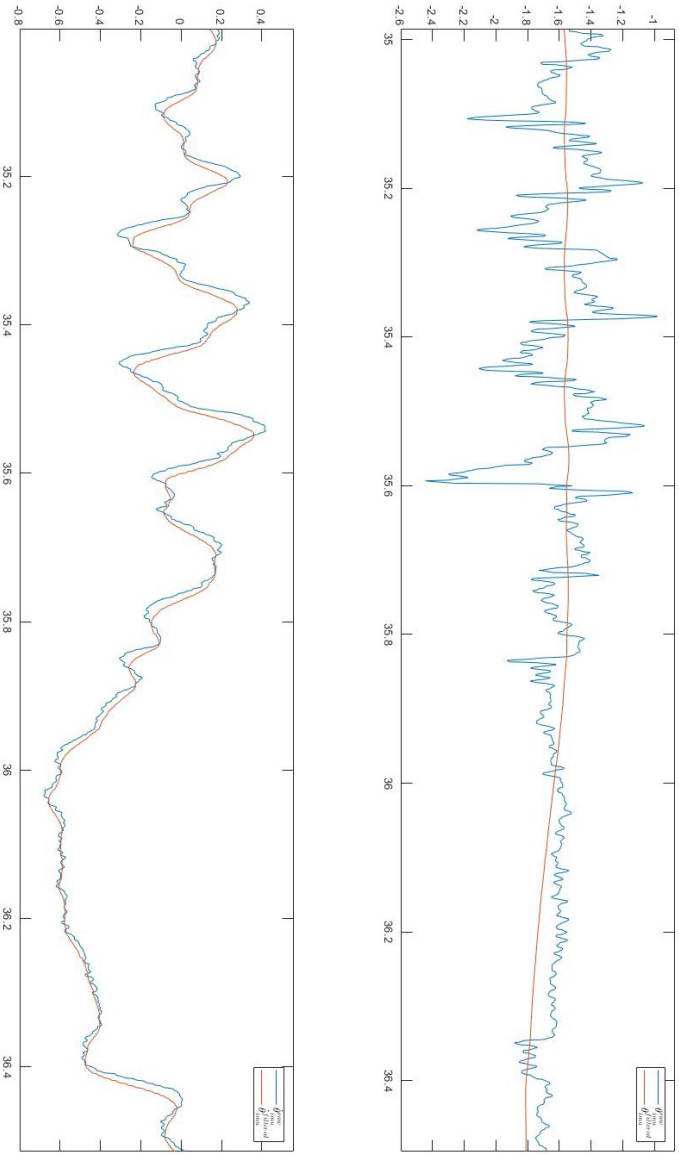

Figure A.2: More Detailed IMU Data (rad vs s)

REFERENCES

[1] A. Lafih and N. Meer, "An overview of rescue robots," 2011.

[2] M. Zafar and H. I. Christensen, "Whole body control of a wheeled inverted pendulum humanoid," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 89–94.

[3] M. Zafar, S. Hutchinson, and E. A. Theodorou, *Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids*, 2018. arXiv: 1810.03074 [cs.RO].

[4] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[5] ——, "Visual servo control. ii. advanced approaches [tutorial]," *IEEE Robotics Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.

[6] J. B. and, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 6, pp. 869–880, Dec. 1996.

[7] W. J. Wilson, C. C. Williams Hulls, and G. S. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 684–696, 1996.

[8] K. Nakao, K. Kondo, S. Kobashi, Y. Hata, T. Yagi, and T. Hayashi, "Object position/-pose estimation using cad models for navigation of manipulator with a single ccd camera," in *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No.03EX694)*, vol. 3, 2003, 1433–1438 vol.3.

[9] K. Wang, Fuquan Dai, Jize Li, and Shaofeng Zeng, "Visually servoed pickup of stationary objects with a kinematically controlled manipulator," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, 2016, pp. 251–256.

[10] H.-U. Ha, H.-N. Yoon, Y.-K. Kim, D.-H. Lee, and J.-M. Lee, "Object tracking of the mobile robot using the stereo camera," in *Intelligent Robotics and Applications*, H. Liu, N. Kubota, X. Zhu, R. Dillmann, and D. Zhou, Eds., Cham: Springer International Publishing, 2015, pp. 322–330, ISBN: 978-3-319-22873-0.

[11] V. Lippiello, B. Siciliano, and L. Villani, "Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 73–86, Feb. 2007.

[12] A. R. Deris, I. Trigonis, A. Aravanis, and E. K. Stathopoulou, "Depth cameras on uavs: A first approach," 2017.

[13] T. Westfechtel, K. Ohno, B. Mertsching, D. Nickchen, S. Kojima, and S. Tadokoro, "3d graph based stairway detection and localization for mobile robots," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 473–479.

[14] W. Ye, Z. Li, C. Yang, J. Sun, C.-Y. Su, and R. Lu, "Vision-based human tracking control of a wheeled inverted pendulum robot," *IEEE Transactions on Cybernetics*, vol. 46, pp. 1–1, Oct. 2015.

[15] H. Wang, A. Chamroo, C. Vasseur, and V. Koncar, "Stabilization of a 2-dof inverted pendulum by a low cost visual feedback," in *2008 American Control Conference*, 2008, pp. 3851–3856.

[16] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, G. Guennebaud, J. Levine, A. Sharf, and C. Silva, "A survey of surface reconstruction from point clouds," *Computer Graphics Forum*, p. 27, 2016.

[17] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen, "On benchmarking camera calibration and multi-view stereo for high resolution imagery," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8.

[18] P. Ulises, C. Sohyung, Asfour, and Shihab, "Volumetric calibration of stereo camera in visual servo based robot control," *International Journal of Advanced Robotic Systems*, vol. 6, no. 1, p. 5, 2009. eprint: `https://doi.org/10.5772/6767`.

[19] A. Burbano, M. Vasiliu, and S. Bouaziz, "3d cameras benchmark for human tracking in hybrid distributed smart camera networks," in *Proceedings of the 10th International Conference on Distributed Smart Camera*, ser. ICDSC '16, New York, NY, USA: ACM, 2016, pp. 76–83, ISBN: 978-1-4503-4786-0.

[20] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "Dart: Dynamic animation and robotics toolkit," *J. Open Source Software*, vol. 3, p. 500, 2018.

[21] N. Dantam and M. Stilman, "Robust and efficient communication for real-time multiprocess robot software," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, 2012, pp. 316–322.

[22] L. Barinka and I. R. Berka, “Inverse kinematics - basic methods.”

[23] S. Feng, “Online hierarchical optimization for humanoid control,” 2016.

[24] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[26] K. Li, T. Pham, H. Zhan, and I. D. Reid, “Efficient dense point cloud object reconstruction using deformation vector fields,” in *ECCV*, 2018.

[27] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4, 2011.

[28] S. G. Johnson, “The nlopt nonlinear-optimization package.”

[29] K. Svanberg, “A class of globally convergent optimization methods based on conservative convex separable approximations,” *SIAM Journal on Optimization*, pp. 555–573,

[30] M. Zafar, A. Mehmood, M. Khan, S. Zhang, M. Z. Murtaza, V. Aladele, E. A. Theodorou, S. A. Hutchinson, and B. Boots, “Semi-parametric approaches to learning in model-based hierarchical control of complex systems,” 2018.

[31] M. Bak, *Control of Systems with Constraints*. Technical University of Denmark: Ørsted-DTU, Automation, 2000.

[32] K. J. Astrom and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ, USA: Princeton University Press, 2008, ISBN: 0691135762, 9780691135762.

[33] H. Wang, A. Chamroo, C. Vasseur, and V. Koncar, “Hybrid control for vision based cart-inverted pendulum system,” Jul. 2008, pp. 3845 –3850.

[34] S. Parsons, “Probabilistic robotics by sebastian thrun, wolfram burgard and dieter fox, mit press, 647 pp., $55.00, isbn 0-262-20162-3,” *Knowl. Eng. Rev.*, vol. 21, no. 3, pp. 287–289, Sep. 2006.